Task - 1 : Eating Out

Introduction :

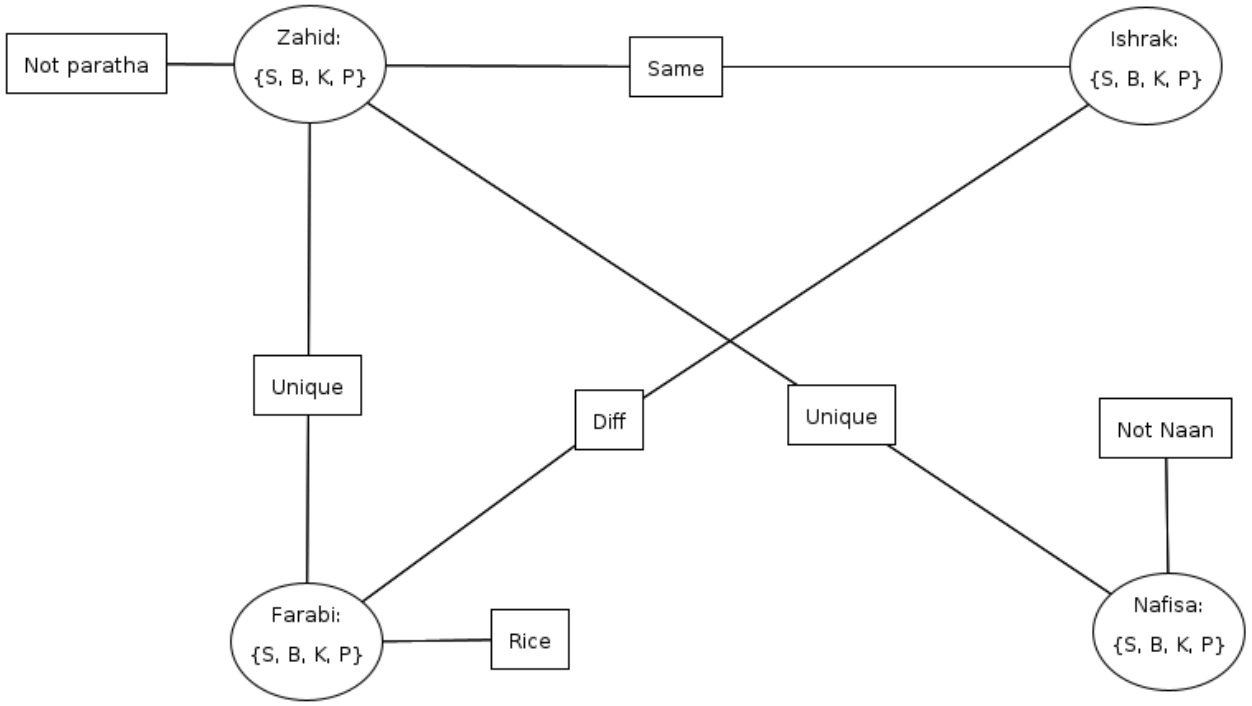
Zahid (Z), Ishrak (I), Farabi (F), and Nafisa (N)'s preferences for food and dependency on choices are given. We need to formulate the CSP which is solvable by the CSP solver satisfying all the constraints.

Problem Analysis :

Variables : Zahid, Ishrak, Farabi, Nafisa (Z, I, F, N)
Domains : Special Rice, Biriyani Rice, Kashmiri Naan, Paratha - { S, B, K, P }
Constraints :

- Zahid Does not like Paratha (Z != P (Paratha))
- Ishrak and Farabi's food should be different (I!= F)
- Farabi prefers Rice Item { S or B }
- Zahid wants to be unique but may copy Ishrak (Z = I) and (Z != F, Z!=N)
- Nafisa won't take Kashmiri Naan : N!=K

Formulation of the CSP :



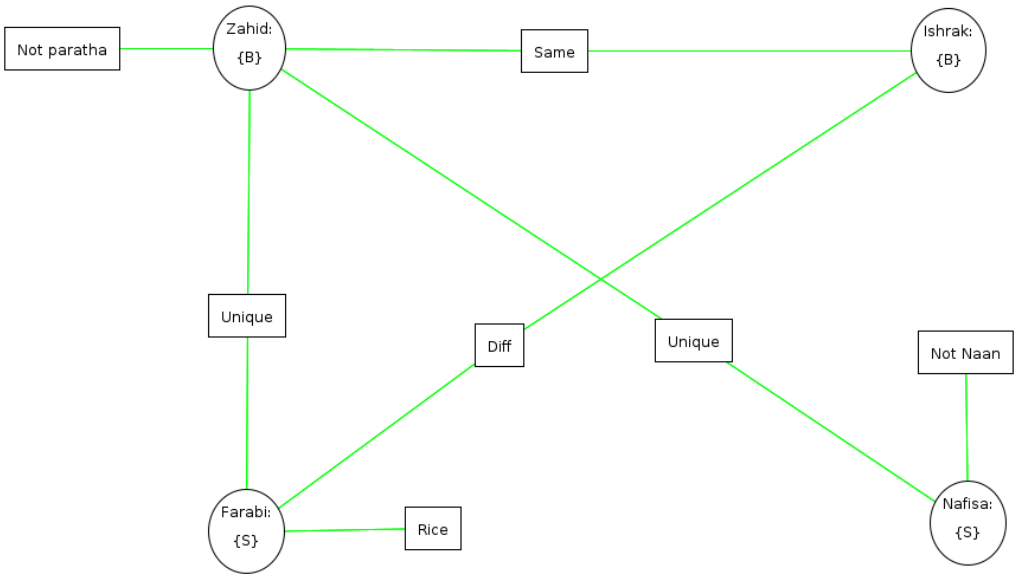
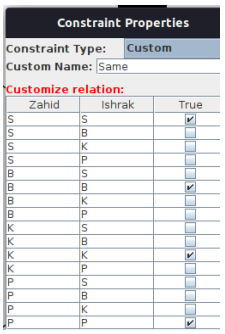


Figure: Formulation and Solution of the CSP for Task - 1

Solution Explanation :

In the formulated CSP, Circles are the Variables and the curly braces in the circle contain domains (Strings - S, B, K, P) for that variable.

- Arcs between circles are the binary constraints between variables. For Example : Z = I, Z! = F, Z! = N etc. Binary constraints define the relation between two distinct variables. I used truth table-based validation while setting up the constraints.



Constraint Properties

Constraint Type: Custom

Custom Name: Same

*Customize relation:

Zahid	Ishrak	True
S	S	☑
S	B	☐
S	K	☐
S	P	☐
B	S	☐
B	B	☑
B	K	☐
B	P	☐
K	S	☐
K	B	☐
K	K	☑
K	P	☐
P	S	☐
P	B	☐
P	K	☐
P	P	☑

- The outgoing arcs from one circle that is not connecting another circle are the unary constraints. For Example : Z ! = P, N! = K, F = {S,B} etc .
  Here, a unary constraint defines the set of domains a variable can take or can't take.
- Running the CSP Solver, we can see that we found a solution for the formulated CSP.

```
DOMAIN-SPLITTING HISTORY:

Farabi in {S}
   Zahid in {B}
      Nafisa in {S}
         Solution found:  Zahid = B, Ishrak = B, Farabi = S, Nafisa = S
```

And the solution meets the constraints of the given problem. Also, there can be multiple valid solutions in this case. So we found one of the solutions in the solver.

Challenges :

No challenge was faced in this particular task as it was from one of the mid-questions.

Task - 2 : Finding Houses

Introduction :

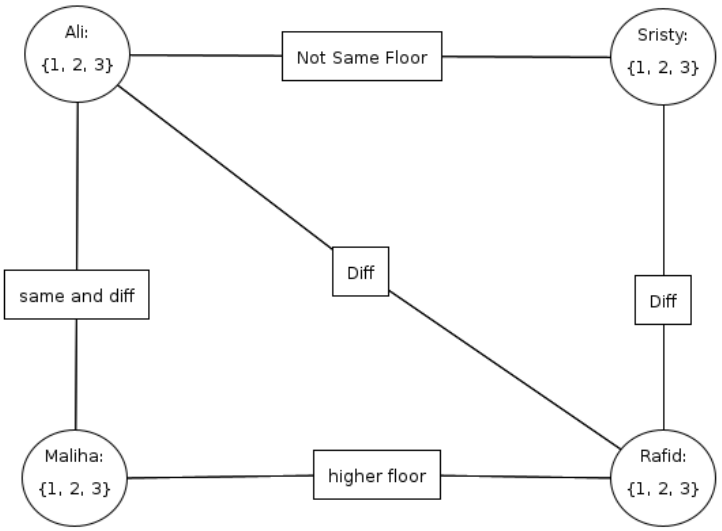
We need to formulate a CSP that assigns 3 floors of a building to 4 people maintaining the given constraints.

Problem Analysis :

Variables : Ali (A), Sristy (S), Maliha (M), Rafid (R)
Domains : { 1, 2, 3 } - Floors of the building
Constraints :

- A and S must not live on the same floor (A!=S)
- If A and M in the same floor, they must be in 2 (A = M = 2)
- If A and M are different, one of them in 3 (A!=M and (A or M = 3))
- R cannot share the floor with anyone. (R!=M, R!=A, R!=S)
- R lives on a higher floor than M (R > M)

Formulation of the CSP :



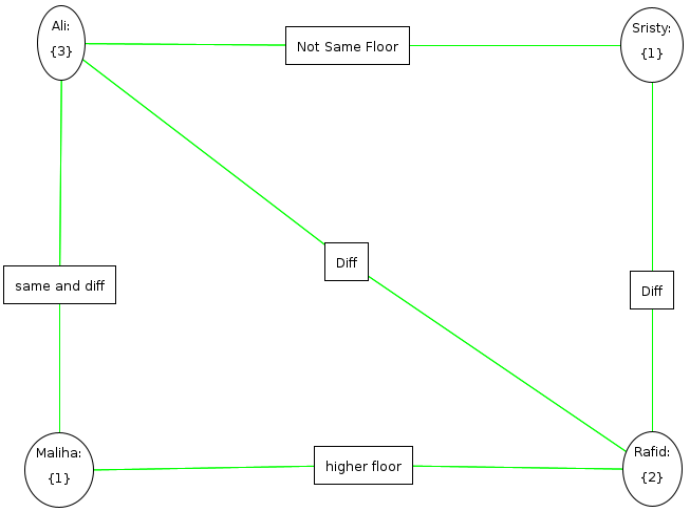
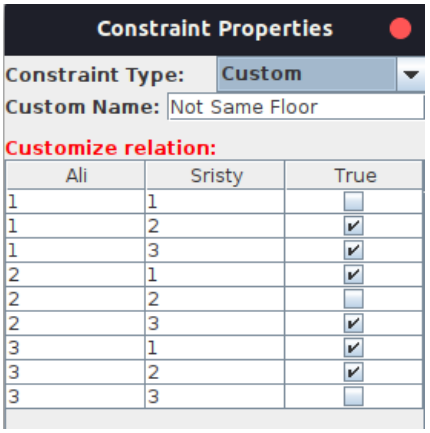


Figure: Formulation and Solution of the CSP for Task - 2

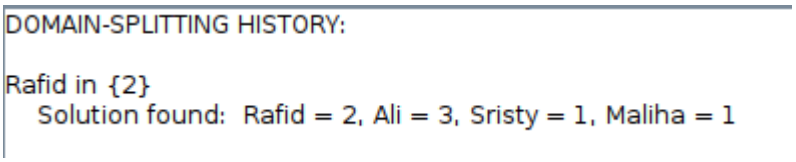
Solution Explanation :

In the formulated CSP, Circles represent the variables (A, R, M, S), and the curly braces inside them contain the domain for that variable.
Here, domain refers to all possible values a variable can take on. The floors of the building are considered as domain.
So the type of the domain is integer {1, 2, 3}.
Since all the constraints given relate to two variables at least, so we have only binary constraints for this problem.

For this problem, I used the truth table-based constraint type for setting up all binary constraints.



Running the CSP Solver, we can see that a solution was found for the formulated CSP



In the solution, we can see, that Rafid is on a higher floor than Maliha and also he isn't sharing the floor with anyone. Ali and Maliha in different floors and Ali is on floor 3.
So, the solution meets the constraints of the given problem. Also, there can be multiple valid solutions in this case. So we found one of the solutions in the solver.

Challenges Faced :

No challenge was faced in this particular task as it was from one of the mid-questions.

Findings :

I could use equality-based constraints while setting up the binary constraints. It could reduce the work of manually setting values in the truth table.

Task - 3 : Spots

Introduction :

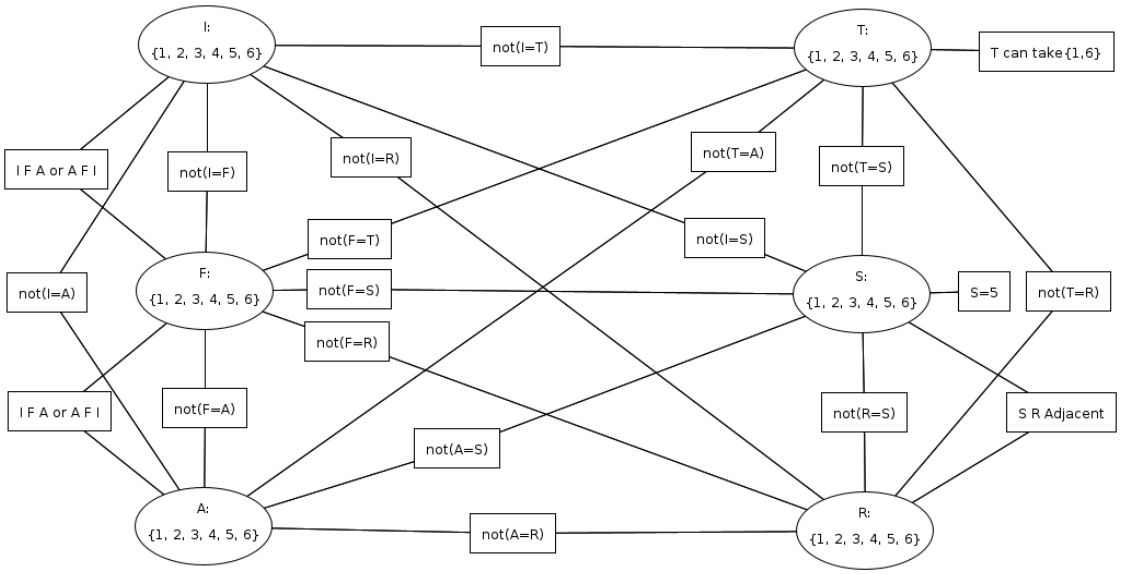
Six Friends are standing in a queue. There are some constraints on which positions they can stand or not and each of them has to occupy a unique spot in the queue. We need to formulate the CSP according to the constraints.

Problem Analysis:

Variables : Ishmam (I), Farhan (F), Atiq (A), Tabassum (T), Sabrina (S), Rifat (R).
Domain : Position in the queue {1, 2, 3, 4, 5, 6}
Constraints :

- T can take 1 or 6th position (T = {1, 6}
- S has only one person behind her so S = 5
- F is standing between A and I, so the formation will be I - F - A or A - F - I
- All are in the different unique spot - AllDiff(I, F, A, T, S, R)

Formulation of the CSP :



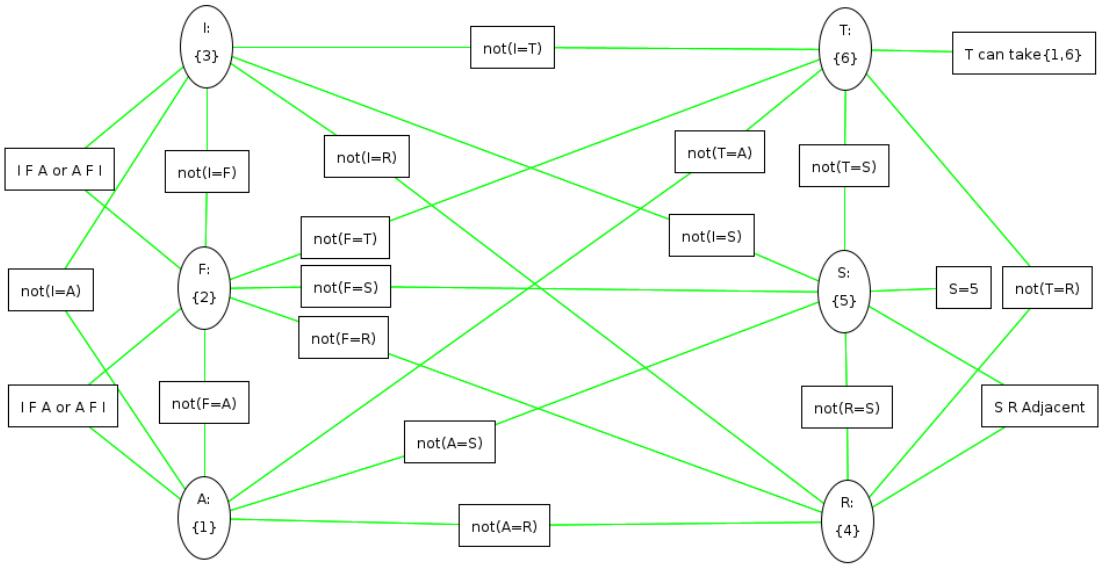
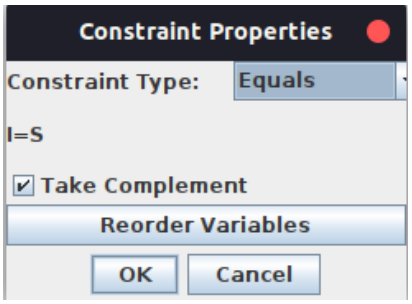


Figure : Formulation and Solution of the CSP for Task - 3

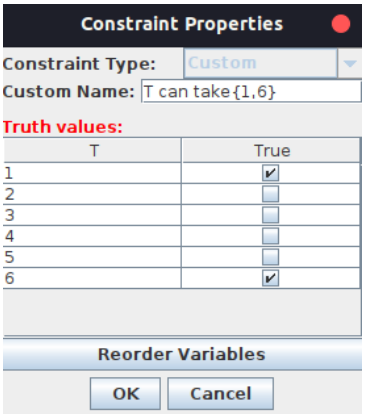
Solution Explanation :

In the formulated CSP, Circles represent the variables (I, F, A, T, S, R), and the curly braces inside them contain the domain for that variable.
Here, domain refers to all possible spots a person can take on. The positions in the queue are considered as domains. So the type of the domain is integer {1, 2, 3, 4, 5, 6}.
To align with the given constraints, I had to take two types of constraints. The constraint between two variables is considered a binary constraint.



To impose the binary constraint: AllDiff(I, F, A, T, S, R), I added 5 binary constraints from each variable to 5 distinct variables which implies each variable has a different value from the other 5 variables. Here I used the inequality constraint as the binary constraint.

I also used truth table-based constraint as a binary constraint in the case of F standing between I and A. Here I took two arcs, one is between I and F and another is between F and A, in the truth table, I selected the values such that $|F-I| = 1$ and $|F-A| = 1$.
In the case of S and R standing next to each other, I also took truth table-based constraint selecting the values that align with $|R-S| = 1$.



I used unary constraints in two scenarios, such as:

- S has one person behind her
- T can take 1st or 6th position.

Running the CSP Solver, we can see that a solution was found for the formulated CSP :

```
DOMAIN-SPLITTING HISTORY:

R in {4}
   T in {1}
      A in {2}
         Cannot split variable R
      A in {3}
         Cannot split variable R
   T in {6}
      A in {1}
         Solution found:  R = 4, A = 1, F = 2, I = 3, T = 6, S = 5
```

And the solution meets the constraints of the given problem.

Challenges Faced: No particular challenges were faced as I used equality constraints in AllDiff(), which I learned after Task - 2. Without it, this might be more hectic.

Task - 4 : Scheduling Tasks

Introduction :

In the problem, we need to schedule 5 tasks (AI Class, AI Lab, Script Checking, DBMS Lab, Content Gathering and each has a different timespan) which is done by two faculty members (X and Y) in a timeframe of 8 AM to 12 AM. We need to formulate the CSP considering the given constraints.

Problem Analysis :

Variables : Q (Script Checking), D (DBMS Lab), C (AI Class), L (AI Lab), G (Resource Gathering)

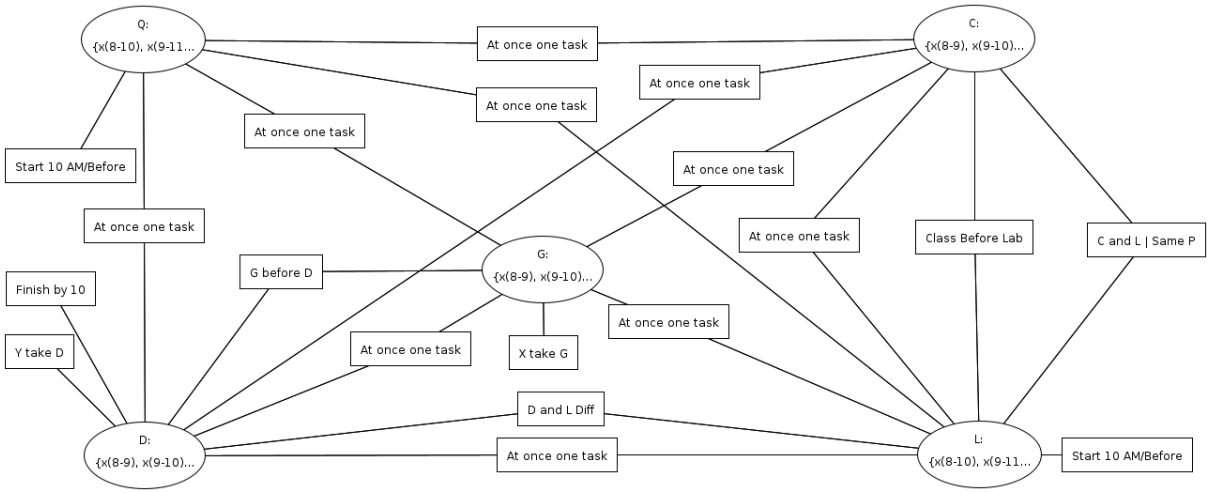
Domains :

{x(8-9), x(9-10), x(10-11), x(11-12),
y(8-9), y(9-10), y(10-11), y(11-12),
x(8-10), x(9-11), x(10-12),
y(8-10), y(9-11), y(10-12)} - Basically all available timeframes. Here x represents the time taken by the faculty member “X” and y represents the time taken by the faculty member “Y”

Constraints :

- X or Y can do one task at a time
- Q must start at or before 10 AM (Duration 2 Hours)
- L must start at or before 10 AM (Duration 2 hours)
- D should be finished by 10 AM
- G must be done before D
- Faculty member X does G and Faculty Member Y takes D
- C must happen before L
- C and L must be taken by the same faculty member.
- D and L must be taken by the different faculty members.

Formulation of the CSP :



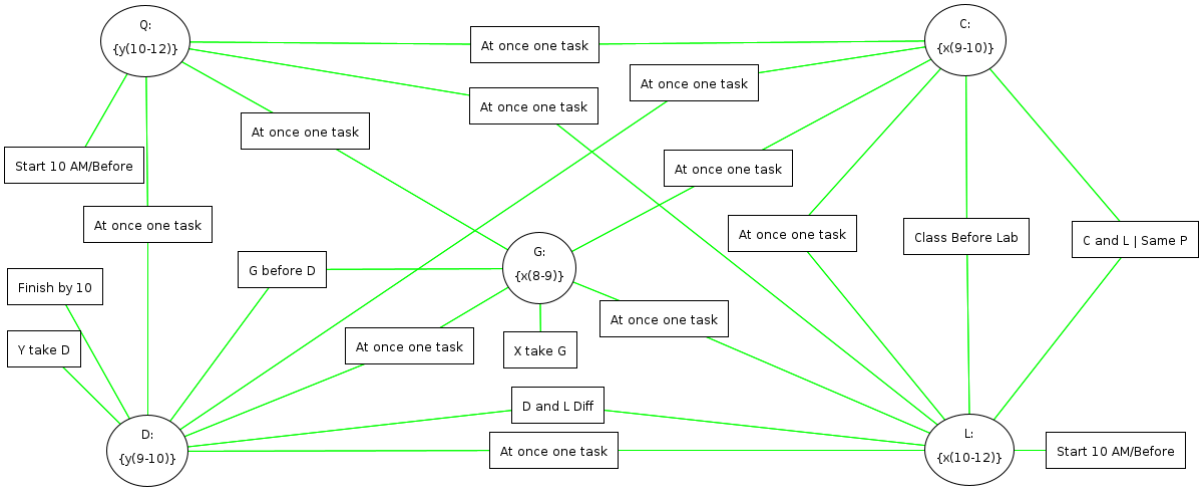


Figure: Formulation and Solution of the CSP for Task - 4

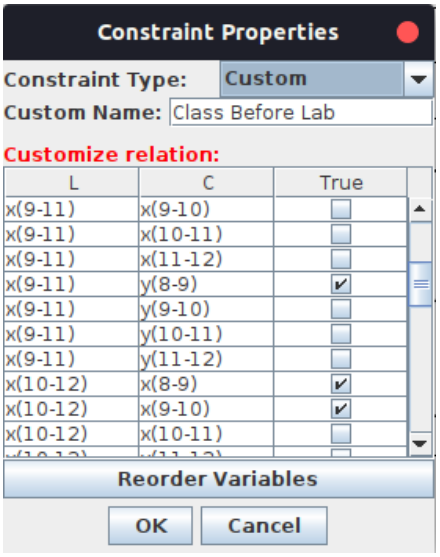
Solution Explanation :

In the formulated CSP, Circles represent the variables (Q, D, C, L, G) and the curly braces inside them contain the domain for that variable.

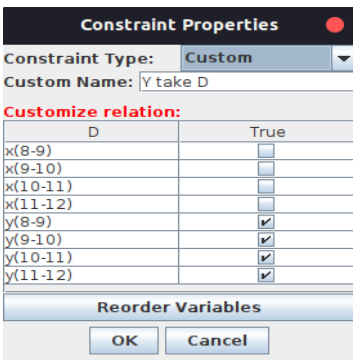
Here, domain refers to the all possible timeframes a particular task can take on depending on the duration of that task. The different possible timeframes of the faculty members are taken as the domain i.e. x(8-9), x(9-10), x(10-11), x(11-12), y(8-9), y(9-10), y(10-11), y(11-12) and so on. So the most suitable type of domain is String.

Boolean or Integer domain types like the previous tasks aren't enough to incorporate the constraints of this problem because we need to consider both faculty members and the timeframes here.

To impose "one task at a time" I used 4 binary constraints from each task to the other 4 tasks by unchecking the matches or overlapping in timeframes. This ensures both faculty members and timeframe constraints so that one faculty member can do only one task at a time.



I also used binary constraints in the cases of C must take place before L, C and L must be taken by the same faculty member, D, and L must be taken by a different faculty member, and G must be done before D. I used Truth table based constraints in those cases.



I needed to impose some unary constraints also like Q must start at or before 10 AM, L must start at or before 10 AM, D should be finished by 10 AM, G must be done before D,

Faculty member X does G, and Faculty Member Y takes D. I've done that by unchecking the inappropriate values for that variable.
Running the CSP Solver, we can see that a solution was found for the formulated CSP :

Q - y{10 - 12}
D - y{9 - 10}
C - x{9 - 10}
L - x{10 - 12}
G - x{8 - 9}

The solution implies, X will gather contents from 8-9 AM and then X will take AI Class from 9 - 10 AM and then AI Lab from 10 - 12 AM
On the other hand, Y will take DBMS Lab from 9 - 10 AM and then check the quiz scripts from 10 - 12 AM which align with the given constraints.

Challenges Faced :

Initially, I faced some difficulties while trying to take binary, and integer domain types like previous tasks. But later understood string is a good fit here as a domain type that is capable of describing and relating all the constraints including faculty members and the timeframes.