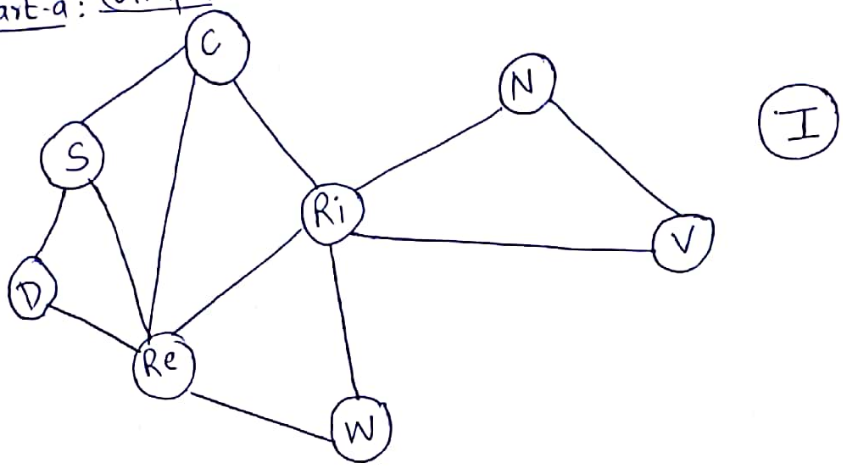
Assignment-2

Name-Yunika Upadhayaya
ID-1001631183

Task-1

Part-a: (Graph)



Part-b: Number of constraints for each state is given as (sorted):
⇓
based on number of constraints

State	Constraints
Ri	5
Re	5
C	3
S	3
N	2
V	2
W	2
D	2
I	0

Note: We select the state with least remaining value in Minimum Remaining Value (MRV) and we select the state with highest number of constraints in Degree Heuristic.

Following variables will be selected in each level:

Level 0: We select Ri based on the maximum number of constraints.

level 1: We have 5 variables (Re, W, C, V, N) with MRV. We select Re based on maximum number of constraints.

level 2: We have ~~4~~ variables (C, S, W, D) with MRV. So, we select C based on maximum number of constraints.

Level 3: We have 3 variables (S, W, D) with MRV. So, we select S based on maximum number of constraints.

Level 4: We have 2 variables (W, D) with MRV. So, we select W based on maximum number of constraints.

Level 5: We have 1 variable D with MRV. So, we select D based on maximum number of constraints.

Level 6: We have two variables (N, V) with MRV. So, we select N based on number of constraints.

level 7: We have on variable V with MRV. So, we select V based on number of constraints.

level 8: We have one variable I with MRV. So, we select I based on number of constraints.

Part-c: Assigning 'Red' to the first value selected in part-b and all other variables using Arc consistency:

Step-1:

Ri → R

Re → G (Assume 'G' to Re)

we will check what can we assign to other states using arc consistency. So, at first:

Initial state:

C → ~~R~~ ~~G~~ B

S → R ~~G~~ ~~B~~

N → ~~R~~ G B

V → ~~R~~ G B

W → ~~R~~ ~~G~~ B

D → ~~R~~ ~~G~~ B

I → R G B

Step-2 Checking for arc consistency of Re:

~~C → Re~~ (Re is green, so C cannot be green) → new arcs added
~~S → Re~~ (Re is green, so S cannot be green) → new arcs added
~~D → Re~~ (Re is green, so D cannot be green) → new arcs added
~~Ri → Re~~ (Re is green, Ri is Red) → no new arcs
~~W → Re~~ (Re is green, so W cannot be green) → new arcs added

→ Adding arcs with end point C:
~~Re → C~~ (Re is green, C can be red or blue) → no new arcs
~~Ri → C~~ (Ri is red, C cannot be red) → new arcs added
~~S → C~~ [C is blue (not red or green), so S cannot be blue] → new arcs added

→ Adding ars with end point S:
(S cannot be red) → new arcs added
~~D → S~~ [S is red, so D cannot be red) → new arcs added
~~Re → S~~ [S is red, Re is green) → no new arcs added
~~C → S~~ (C is blue, S is red) → no new arcs added

→ Adding arcs with end point D:
~~Re → D~~ (Re is green, D can be red or blue) → no new arcs
~~S → D~~ (S is red, D can ~~not~~ be blue) → no new arcs added

→ Adding arcs with end point W:
~~Re → W~~ (Re is green, W can be red or blue) → no new arcs
~~Ri → W~~ (Ri is red, W canot be red) → arcs added

→ Adding arcs with end point Ri:
~~C → Ri~~ (C is blue, Ri is red) → no new arcs
~~Re → Ri~~ (Re is green, Ri is red) → no new arcs
~~W → Ri~~ (W is blue, Ri is red) → no new arcs
N → Ri (W cannot be red) → arcs added — (Ri is red, N cannot be red) → arcs added
V → Ri (Ri is red, V cannot be red) → arcs added

→ Adding arcs with end point S:
~~D → S~~ (D is blue, S is Red) → no new arcs
~~Re → S~~ (Re is green, S is Red) → no new arss
~~C → S~~ (C is blue, S is Red) → no new arcs

→ Adding end points N:
~~V → N~~ (V & N can be either green or blue) → no new arcs
~~Ri → N~~ (Ri is red, N can be green or blue) → no new arcs

→ Adding arcs with end point V:
~~N → V~~ (V & N can be either green or blue) → no new arcs
~~Ri → V~~ (Ri is red, V can be green or blue) → no new arcs

step-3: checking for arc consistency of N:

present state:

Ri → R

Re → G

C → B

S → R

N → G~~B~~

V → ~~G~~B

W → B

D → B

I → RGB

let's assume N to be Green:

Arcs with end points N:

~~Ri → N~~ (Ri is Red, N can be green) → no new arcs

~~V → N~~ (N is green, V cannot be green (i.e. should be blue) → arcs added

→ Arcs with end points V:

~~N → V~~ (N is green, V is blue) → no new arcs

~~Ri → V~~ (Ri is red, V is blue) → no new arcs

step 4: Present state:

Ri → R

Re → G

C → B

S → R

N → G

V → B

W → B

D → B

I → R~~G~~~~B~~

"I" can be red, since it is not connected to any other states.

Step-5: Final color of states:

Ri → R

Re → G

C → B

S → R

N → G

V → B

W → B

D → B

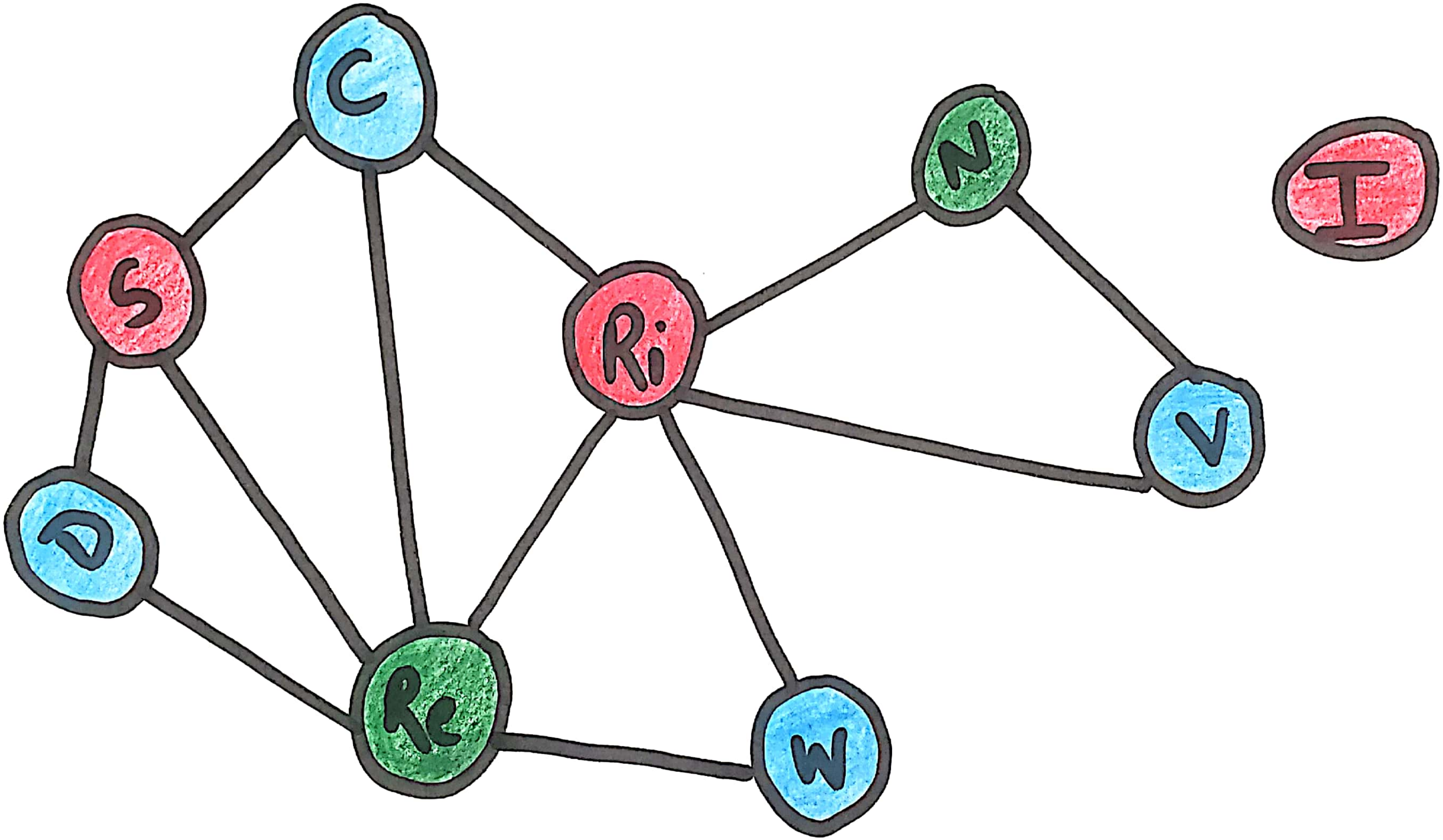
I → R

State	Ri	Re	C	S	N	V	W	D	I
Initial	RGB	RGB	RGB	RGB	RGB	RGB	RGB	RGB	RGB
level 0	R	GB	GB	RGB	GB	GB	GB	RGB	RGB
level 1	R	G	B	RB	GB	GB	B	RB	RGB
Level 2	R	G	B	R	GB	GB	B	RB	RGB
level 3	R	G	B	R	GB	GB	B	B	RGB
Level 4	R	G	B	R	GB	GB	B	B	RGB
Level 5	R	G	B	R	GB	GB	B	B	RGB
Level 6	R	G	B	R	G	B	B	B	RGB
Level 7	R	G	B	R	G	B	B	B	RGB
Level 8	R	G	B	R	G	B	B	B	R
Final	R	G	B	R	G	B	B	B	R

Part-d

Yes, we can use structure of the problem to make solving it more efficient. State "I" is not connected to the mainland graph with other states. Thus, we can conclude that "I" and mainland are independent subproblems. "I" can be colored with any (Red or Green or blue), so we donot have to compute for the color of "I". In this problem, we can check for independence just by checking whether the variables are connected in the graph or not.

Part-e Valid solution to the problem in terms of constraint graph:



Task-2

$x_4 \quad x_3 \quad x_2 \quad x_1$

$$\begin{array}{cccccc} & & S & E & N & D \\ + & & M & O & R & E \\ \hline & M & O & N & E & Y \end{array}$$

Variables: S E N D M O R Y X_1 X_2 X_3 X_4

Domains: $\{0,1,2,3,4,5,6,7,8,9\}$

Constraints: $M \neq 0$; $S \neq 0$

$S \neq E \neq N \neq D \neq M \neq O \neq R \neq Y$

$D + E = 10X_1 + Y$

$N + R + X_1 = E + 10X_2$

$E + O + X_2 = N + 10X_3$

$S + M + X_3 = O + 10X_4$

$X_4 = M$

Constraint Graph: