CSP

Constraint satisfaction problem

Formal Definitation :-

Combination of 3 tuple

① X = A set of variables → each represents an unknown value to be determined

② D = a set of domains (finite on infinite) → each domain represent the possible value that a variable can take

③ C = a set of constraints that specify restrictions on allowed combinations of values for subset of variables → each constraints defines a subset of variables and the relationship that must hold among their values

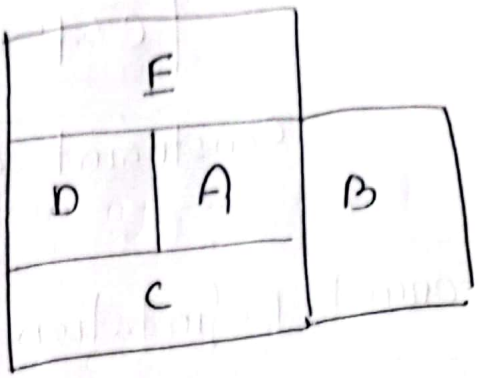
CSP goal is to find best/consistent solution

↓ what is solution ?

Solution is an assignment of values to variables that satisfies all the constraints

Q Map coloring

Colon the following the map using three colons (red, green, blue) such that two adjacent regions donot have same colon.



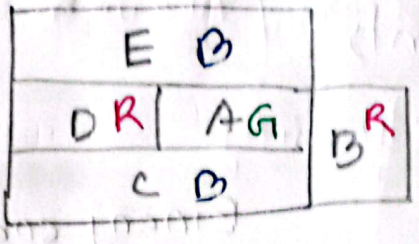
Ans :- CSP

① Variables = {A, B, C, D, E}

② Domains = {Red, Green, Blue} (RGB)

③ Constraints = {A≠B, A≠C, A≠D, A≠E
B≠C, ~~B≠D~~, ~~B≠E~~,
C≠D, ~~D≠E~~, ~~C≠E~~
D≠E}

Based on graph



Solve

A = Green
B = Red
C = Blue
D = Red
E = Blue

CN Constaint Network

→ also know as constraint graph

→ is a graphical representation of CSP

→ provides visualizepresentation of domain, vanicble and constaints.

Two types of constraints

Unary Constraints	Binary constraints
① unary constraints involves one single variable	② Binary constraints involves two variables.
② specify the restriction on condition on the value that a single variable can take.	② It specifies restriction on condition on the value that apair can take
③ It impose constraints on single variable without considering any relation ship with other variable	③ It define relationship on dependency between two variable
④ suppose X = number of hour spent playing	④ suppose, X = number of apple Y = number of orange
∴ unary constraint can be $X \leq 10$	∴ Binary constraints can be $X + Y = 5$

Q Given a list of course to be taught, class room available, time slots and professor who teach certain courses.

can U schedule the class?

(1) Variables :- course offered,
classroom,
time

(2) Domain :- professor,
room number,
time slots

(3) Constaints :-
professor cannot teach more than 2 class,

professor cannot teach 2 class in same timeslot,

1 class per room in each time slot

Systematic search Algorithm

Backtracking

(1) Select an unassigned variable from the set of vanicbles of csp

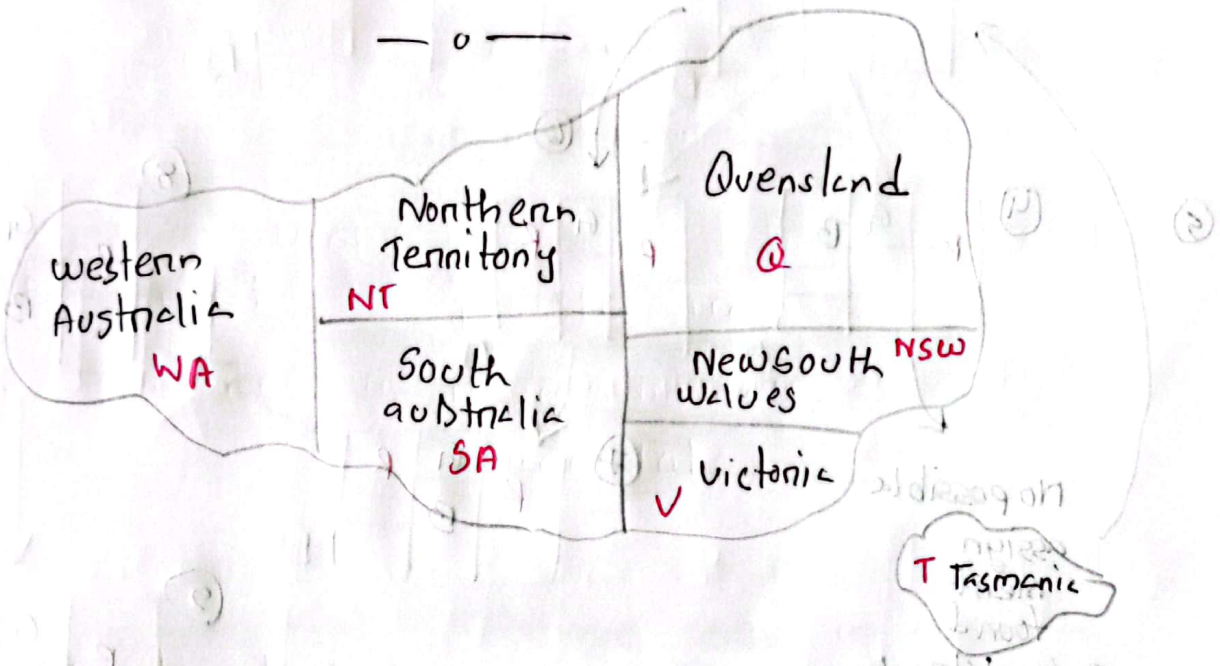
(2) Assign a value from the domain of the selected vanicble.

(3) check constraints :- check if the assigned value violates any constraints with previously assigned vanicbles

(4) Repeat :- If no constraints violated, move to next unassigned value repeat 2, 3, 4 untill a solution is found or no valid assignment is possible

(5) Backtrack :- if no valid assignment possible for current value backtrack to pervious value and undo its assignment. Go back to step 2, 3, 4 with different value.

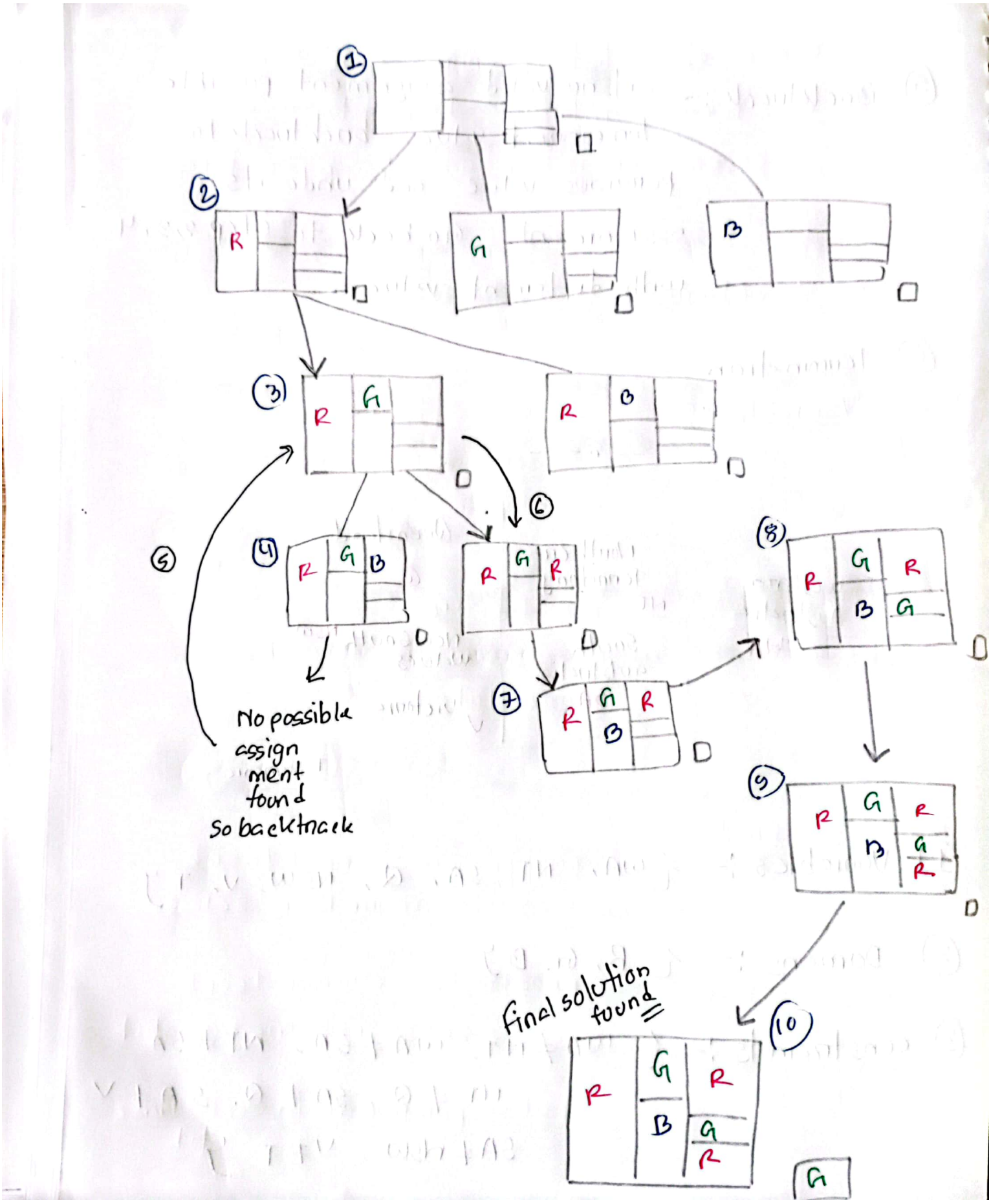
(6) Termination



(1) Vanicables :- {WA, NT, SA, Q, NSW, V, T}

(2) Domains :- {R, G, B}

(3) constraints :- { $WA \neq NT$, $WA \neq SA$, $NT \neq SA$
$NT \neq Q$, $SA \neq Q$, $SA \neq V$
$SA \neq NSW$, $V \neq T$ }


1
2
R
G
B
3
R
G
R
B
4
R
G
B
5
6
R
G
R
7
R
G
R
B
8
R
G
R
B
G
9
R
G
R
B
G
R
10
Final solution found
R
G
R
B
G
R
G
No possible assign ment found So backtrack

Problem of Backtracking

① Thrashing → keep repeating same failed variable assignment

② Inefficient → explore areas that don't lead to success

Solve:-

① use consistency

② Intelligent Backtracking

solve variable ordering

Consistency / constraints propagation

is a fundamental technique used in csp to reduce the search space and improve the efficiency of solving the problem.

Goal: identify and eliminate values from domain of variable that cannot part of valid solution

① Arc consistency

② Node consistency

Variable and value Ordering heuristic

Heuristic can help to make "right" choice for picking and setting the next variable

1. Minimum Remaining Value Heuristic (Most constrained variable)

2. Degree Heuristic

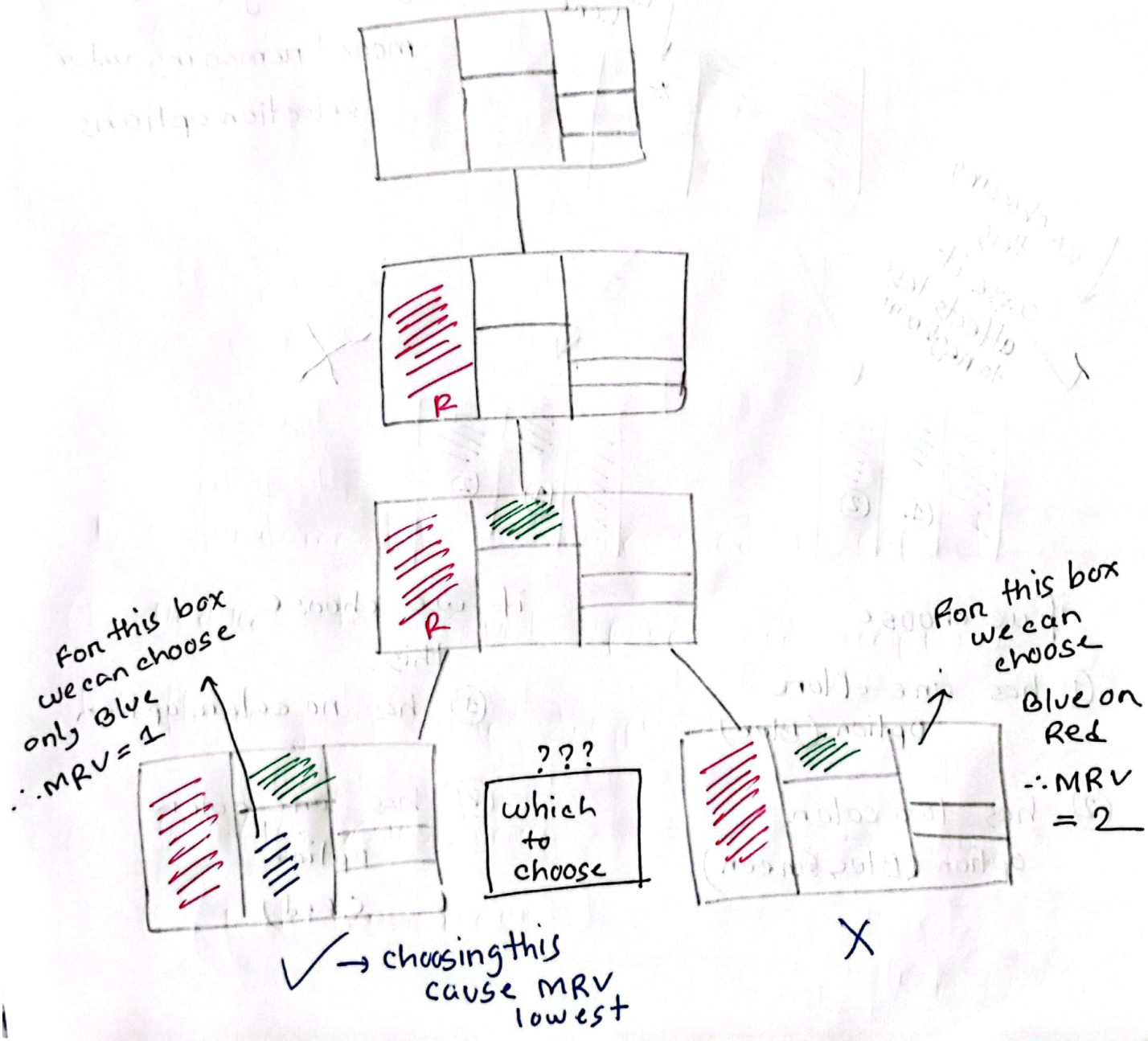
3. Least constraining value heuristic

① most constrained variable

(minimum remaning values heuristic)

MRV

→ choose the value that has smallest number of remaining values in its domain



constraining
constrained

Least ~~conf~~ value

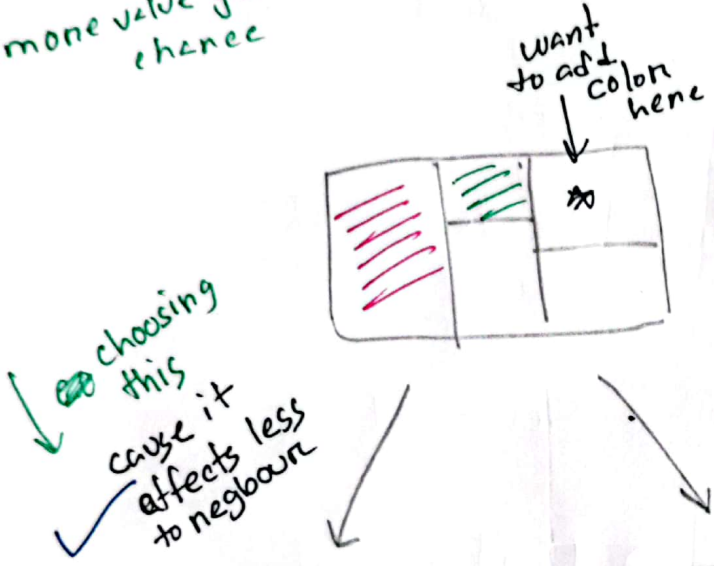
maximum flexibility

Heuristic

→ choose the least constraining value that restricts the remaining variable domain least.

less constraints
- more value getting chance

choose a ~~vale~~ value that gives neighbours more remaining value selection options



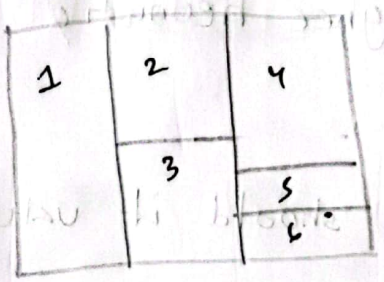
if we choose
① has one colon option (Blue)
② has two colon option (Blue, Green)

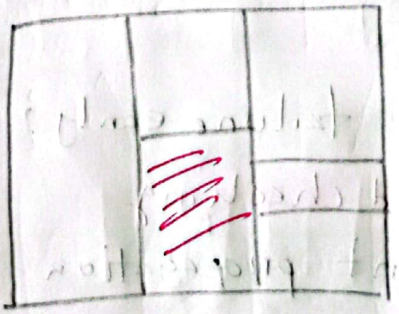
if we choose this
① has no colon option
② has one colon option (Red)

Degree Hunistic → help to select first variable

→ choose a vanicble that is part of most the remaning unsatisfied constraints.

(# pick the vaniable that involved in most constraints) on vaniables





first choose cause most involvement

No involes with other vaniable

1 → 2 (2, 3)

2 → 3 (1, 3, 4)

3 → 5 (1, 2, 4, 5, 6) ✓

4 → 3 (2, 3, 5)

5 → 3 (3, 4, 6)

6 → 2 (3, 5)

Improve backtracking efficiently

General purpose methods can give huge gains

inspeed

(1) which variable should be assigned next?

(variable selection process)

→ most constrained value

→ Degree heuristic

(2) In what order should it values be tried?

value ordering

→ Least Constraining Value

Heuristic

→ ~~forward checking~~

(3) can we detect inevitable failure early?

→ forward checking

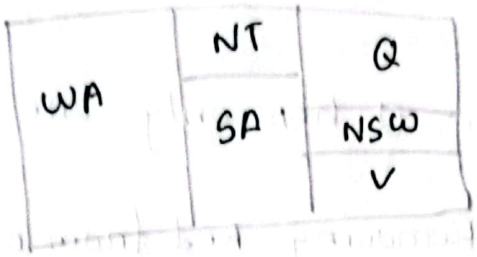
→ constraint propagation

(Arc Consistency)

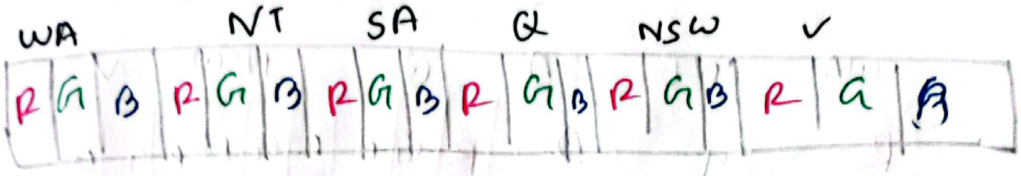
detects failure faster

Forward checking

when assigning a variable, remove the conflicting valved for all connected variable (use back tracking)



step-1

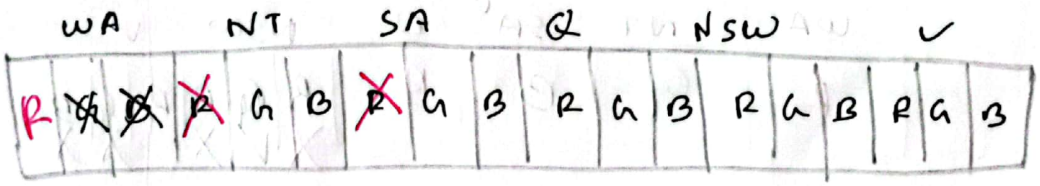


all vanicble has thnee colon option

step-2

Giving WA → Red

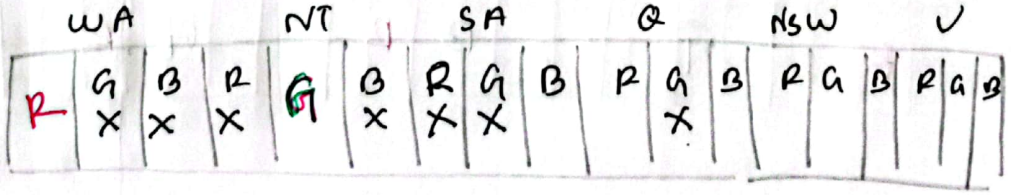
and nemoving Red from adjacent neighbour



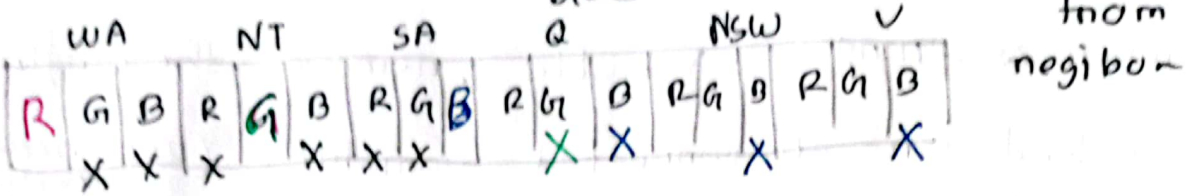
step-3

Giving NT → Gneen

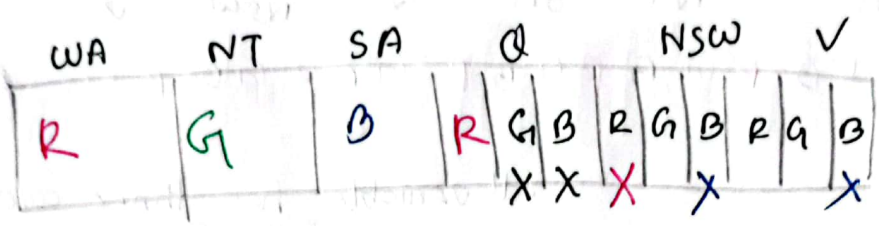
and removing Gneen from adj



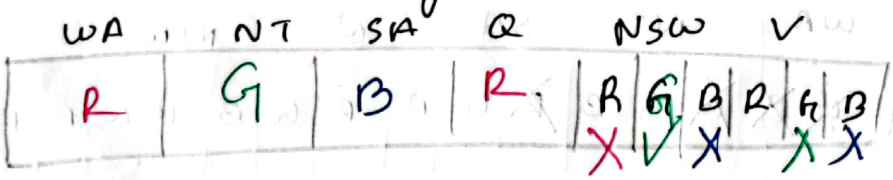
Step-4 :- Now SA has only one option choose blue and removing blue from negibour



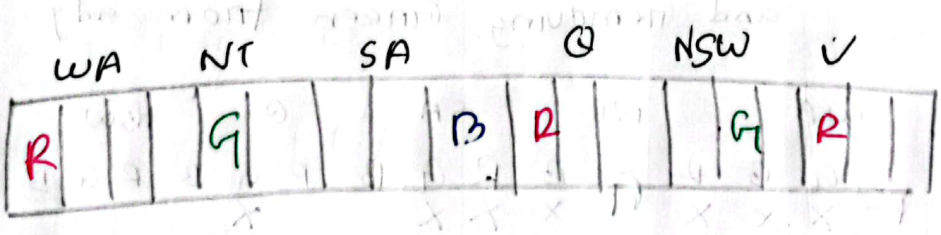
Step-5 ∴ Now Q has only one option (Red) and Removing Red from neighbours



Step-6 NSW has one option Green and removing



Step-7 V got one option Red and solved:



Node Consistency

① a single variable in a csp is node consistent if all values in the variable's domain satisfy the variable's unary constraints.

② Node consistency involves checking the unary constraints for each variable

③ Node consistency is a local property

④ Running node consistency on a node can eliminate all uniary constraints on that node

Arc consistency

① a variable in a csp is arc-consistent if every value in its domain satisfies the variable binary constraits.

② Arc consistency ensures that for each pair of connected variables, every value in the domain of one variable is consisten with some value of domain of another variable

③ Global property

④ To achive arc consistency itenatively examine each constraits and removing values from domain that violates the constraints.

(5) Node consistency doesn't not consider the relationships / constraints between variable.

(6) Example:- we can colon (SA) South africa with Red, Blue, Green. but SA don't like Green.

∴ Domain of SA = {R, G, B}

∴ Unary constaints, SA ≠ G

To make SA node consistent we need remove unary constraints for domain.

As SA don't like green remove green from domain

Now, SA domain = {R, B}

SA is node consistent

(5) Arc consistency consider the constraints between pain of connected variables.

(6) Example

suppose we have two value A

A domain = {1, 2, 3}

B domain = {1, 2}

Binary constraints = A must be greater than the value of B (A > B)

(Goal to remove binary constraints)

For each value in the domain of A we check if there exists a consistent value in the domain of B. If not remove, remove the inconsistent value form domain.

if A = 1 ; No values for B

domain of A = {2, 3}

if A = 2 ; B can have value 1 in domain

domain of A = {3}

if A = 3 ; B can have any value in its domain fufilled the binary constnaits (A > B)

updated domain

A = {2,3}

B = {1,2}

A, B are arc consistent

⑦ a graph is node consistent if all nades are node consistent.

⑦ a graph is arc consistency if all ~~nodes~~ arcs are arc-consistent

—o— Follow this example

① X, Y = {0 - 9}

② $x^2 = y$

Arc consistency

X = {0, 1, 2, 3}

Y = {0, 1, 4, 9}

X are consistent with nespect to Y

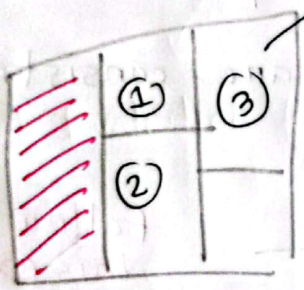
Y are consistenc with nespect X

Arc consistency

→ every value x of X there is allowed y

In arch consistency. if we assign ~~a value~~ an ~~to variable~~, unassigned variable, need to check the neighbours of unassigned variable has an allowable move

(If no allowable move, no assignment)



Green → if we assign the box Green,

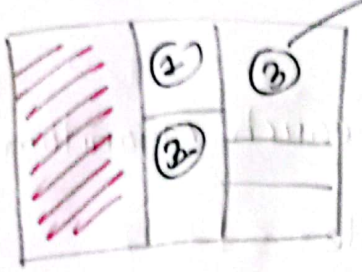
∴ Does its neighboun has next allowable move?

① → Yes ; Blue

② → No allowable move

fails arc consistency

∴ So we will not assign Green to box ③



if we assign ③ → Red

Does its neighboun has an allowable move

① → Blue

② → Green

∴ arc consistency → ✓

∴ Yes Now we can assign ③ → Red