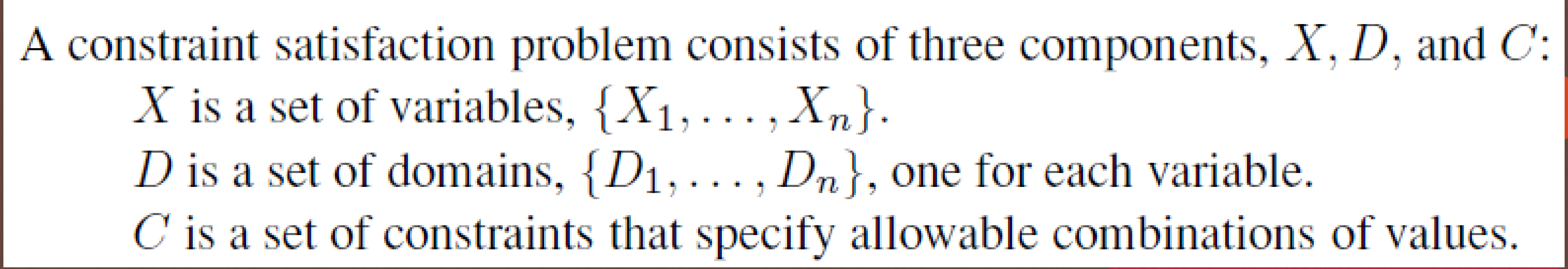
constraint satisfaction problem

Definition

Each state has a value and a variable

Problem gets solved if all the constraints are satisfied by the value of the variable



A constraint satisfaction problem consists of three components, X, D, and C:

X is a set of variables, $\{X_1, \ldots, X_n\}$.

D is a set of domains, $\{D_1, \ldots, D_n\}$, one for each variable.

C is a set of constraints that specify allowable combinations of values.

- D_i has set of values $\{v_{1,}\ldots,v_k\}$ for variable X_i
- Each constraint Ci consists of a pair <scope, rel >
- Scope is a tuple of variables that participate in the constraint
- rel is a relation that defines the values that those variables can take on

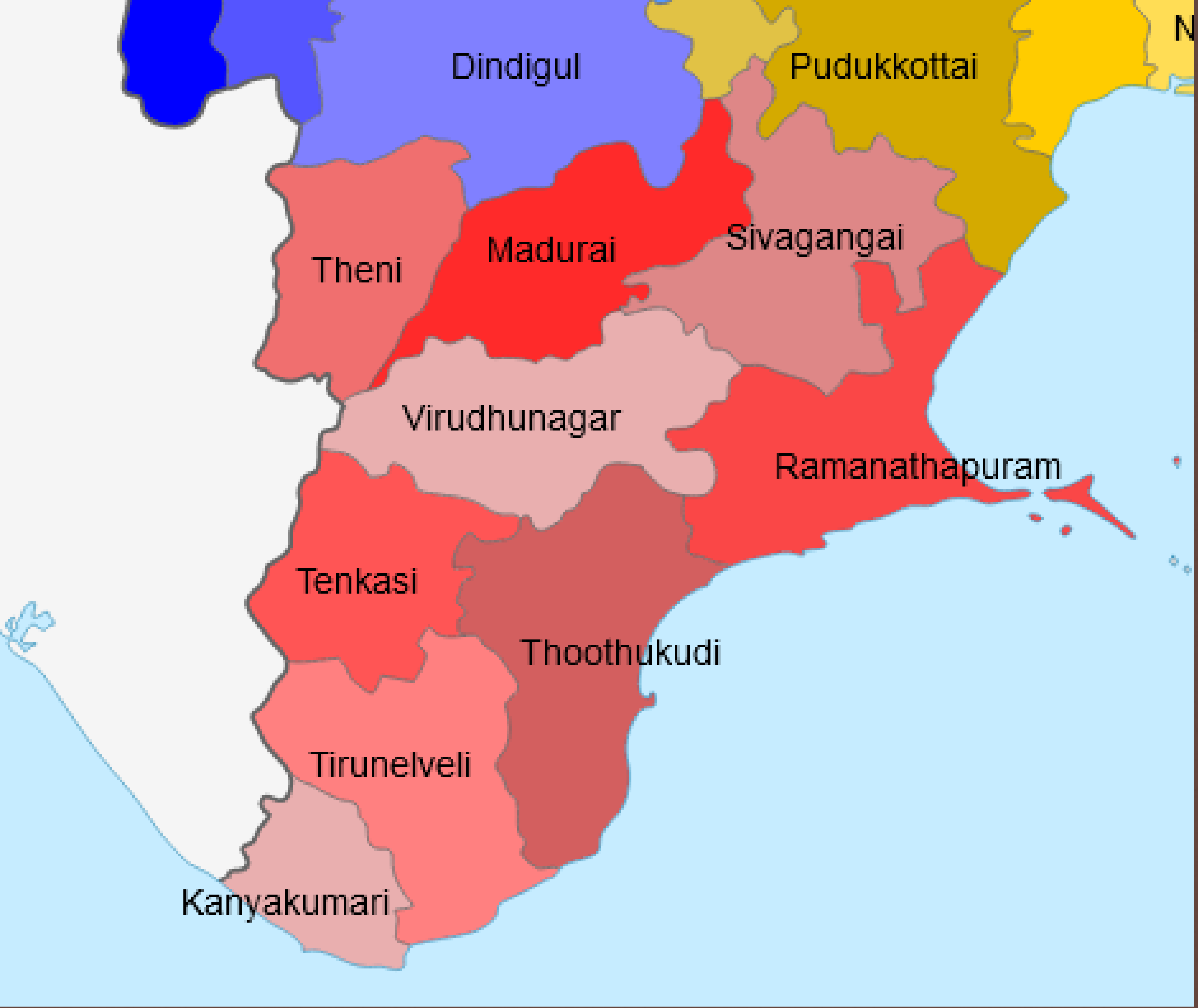
- X1 and X2 has domain A,B then constraint two variable must not have same value is given as

$\langle (X_1, X_2), [(A, B), (B, A)] \rangle$ or as $\langle (X_1, X_2), X_1 \neq X_2 \rangle$

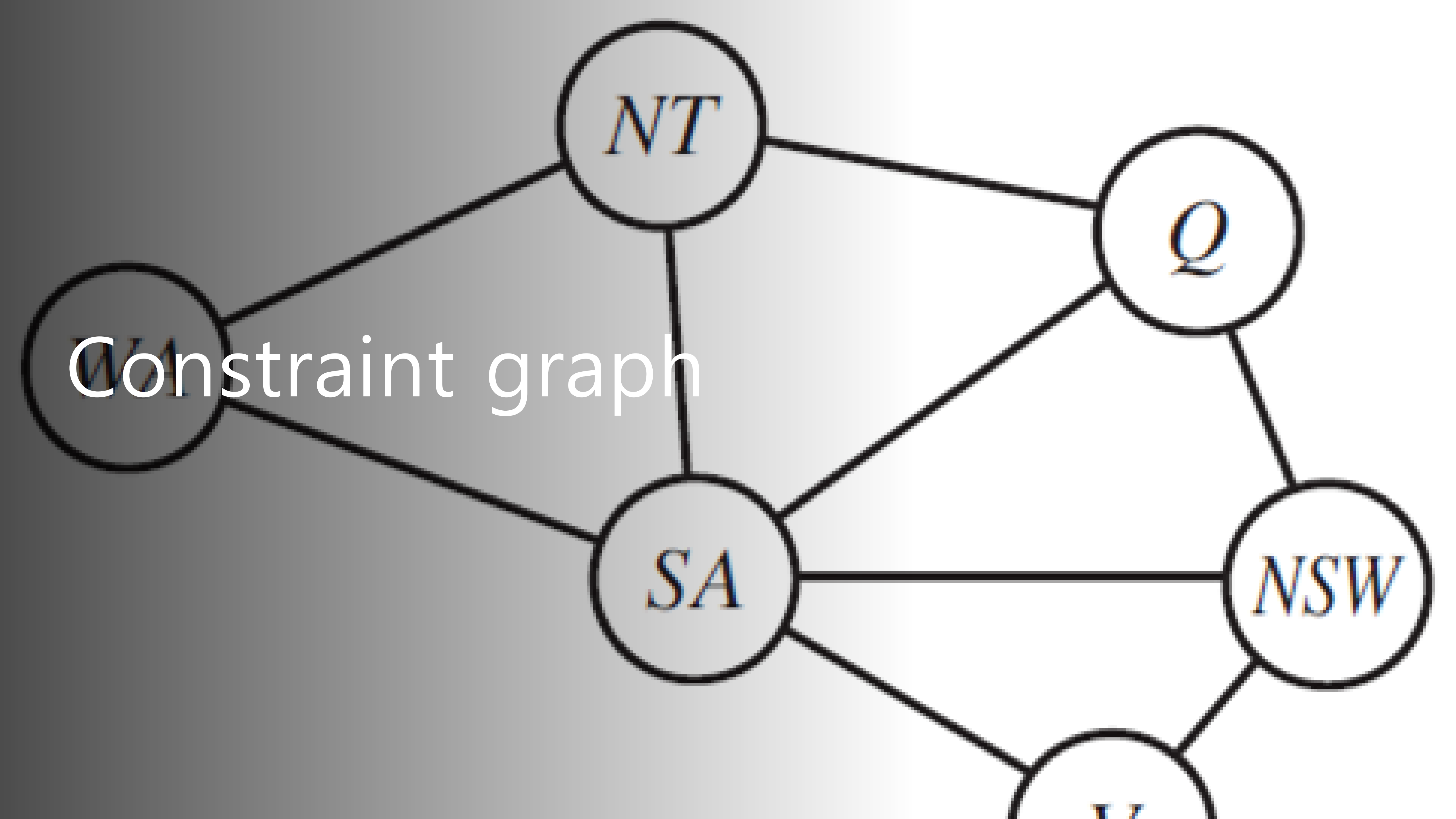
- Assignment - values which are assigned
- Complete assignment – every variables are assigned
- Partial assignment – only a part of variables are assigned

Map colouring problem

- X = {TKSI,TEN,TN,MDU, VM}
- Di = {red,green,blue }
- TKSI ≠ TEN ((TKSI, TEN) , TKSI ≠ TEN)


N
Dindigul
Pudukkottai
Madurai
Sivagangai
Theni
Virudhunagar
Ramanathapuram
Tenkasi
Thoothukudi
Tirunelveli
Kanyakumari

Constraint graph



$$X = \{Axle_F, Axle_B, Wheel_{RF}, Wheel_{LF}, Wheel_{RB}, Wheel_{LB}, Nuts_{RF}, Nuts_{LF}, Nuts_{RB}, Nuts_{LB}, Cap_{RF}, Cap_{LF}, Cap_{RB}, Cap_{LB}, Inspect\} .$$

$$T_1 + d_1 \leq T_2 .$$

$$\begin{aligned}
&Wheel_{RF} + 1 \leq Nuts_{RF}; \quad && Nuts_{RF} + 2 \leq Cap_{RF};\\
&Wheel_{LF} + 1 \leq Nuts_{LF}; \quad && Nuts_{LF} + 2 \leq Cap_{LF};\\
&Wheel_{RB} + 1 \leq Nuts_{RB}; \quad && Nuts_{RB} + 2 \leq Cap_{RB};\\
&Wheel_{LB} + 1 \leq Nuts_{LB}; \quad && Nuts_{LB} + 2 \leq Cap_{LB} .
\end{aligned}$$

Domains

- Discrete, finite domains
- Domain may be infinite [scheduling no time limit]
- Infinite domain constraint cannot be specified
- So use constraint language : $T_1 + d_1 \leq T_2$.

Domains

- Linear constraint as given earlier
- Non linear constraint : there is no algorithm for solving

Types of constraints

- Unary constraints : Restrict value variable [problem of race]
- Binary Constraint [relates two variables]
- Global Constraint : Constraint involving arbitrary number of variables

Types of constraints

- Using constraint reduce the number of legal values
- All inconsistent values are eliminated (through a graph)

Constrain propagation : Node consistency

- Values in the variable’s domain satisfy the variable’s unary constraints.
- Example : race problem, map coloring problem
- Reduce domain {red,green,black} to {red,green}

Arc consistency

- Every value in domain satisfies variable binary constraint
- Eg : $Y = X^2$
- $\langle (X, Y), \{(0, 0), (1, 1), (2, 4), (3, 9)\} \rangle$
- reduce X's domain to {0, 1, 2, 3}
- Y 's domain becomes {0, 1, 4, 9}

Path consistency

- tightens the binary constraints by using implicit constraints that are inferred by looking at triples of variables
- A two-variable set $\{X_i, X_j\}$ is path-consistent with respect to a third variable X_m if, for every assignment $\{X_i = a, X_j = b\}$ consistent with the constraints on $\{X_i, X_j\}$ there is an assignment to X_m that satisfies the constraints on $\{X_i, X_m\}$ and $\{X_m, X_j\}$

K-consistency

- for any set of k − 1 variables and for any consistent assignment to those variables, a consistent value can always be assigned to any kth variable

```
function BACKTRACKING-SEARCH(csp) returns a solution, or failure
  return BACKTRACK({ }, csp)

function BACKTRACK(assignment, csp) returns a solution, or failure
  if assignment is complete then return assignment
  var ← SELECT-UNASSIGNED-VARIABLE(csp)
  for each value in ORDER-DOMAIN-VALUES(var, assignment, csp) do
      if value is consistent with assignment  then
          add {var = value} to assignment
          inferences ← INFERENCE(csp, var, value)
          if inferences ≠ failure then
            add inferences to assignment
            result ← BACKTRACK(assignment, csp)
            if result ≠ failure then
              return result
      remove {var = value} and inferences from assignment
  return failure
```

Inference

- Inference is to impose node, arc , path and k Consistency

Questions for solving CSP efficiently

- Which variable to be assigned next ?
- What inference to be performed ?
- If search violate constraint can it avoid repeating this failure?

Variable and value ordering

- Assign values to the variables
- Remaining values for variables one is chosen and rest with force assignment
- This intuitive idea—choosing the variable with the fewest “legal” values—is called the minimum- remaining-values (MRV) heuristic

Variable and value ordering

- Once a variable has been selected, the algorithm must decide on the order in which to examine its values.
- For this, the least-constraining-value heuristic can be effective
- It prefers the value that rules out the fewest choices for the neighboring variables in the constraint graph

Interleaving search and inference

- Inference - opportunity to infer new domain reductions on the neighboring variables
- Simplest forms of inference is called forward checking
- For each unassigned variable Y that is connected to X by a constraint, delete from Y’s domain any value that is inconsistent with the value chosen for X

Interleaving search and inference

- Forward checking does arc consistency

Intelligent backtracking: Looking backward

- when a branch of the search fails:

back up to the preceding variable and try a different value for it. This is called chronological backtracking

Intelligent backtracking: Looking backward

- A more intelligent approach to backtracking is to backtrack to a variable that might fix the problem
- Back jumping method backtracks to the most recent assignment in conflict set

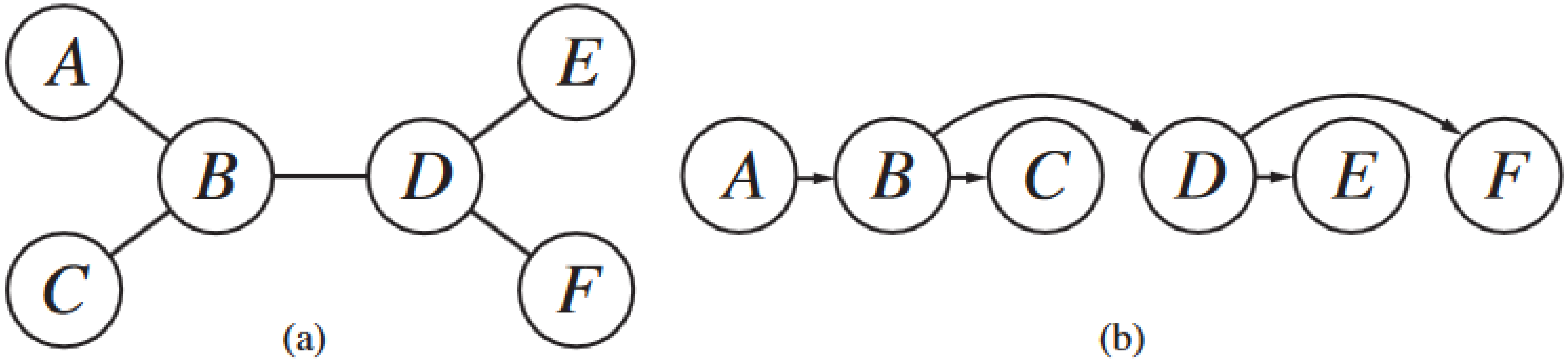
```
function MIN-CONFLICTS($csp$, $max\_steps$) returns a solution or failure
   inputs: $csp$, a constraint satisfaction problem
           $max\_steps$, the number of steps allowed before giving up

   $current \leftarrow$ an initial complete assignment for $csp$
   for $i$ = 1 to $max\_steps$ do
      if $current$ is a solution for $csp$ then return $current$
      $var \leftarrow$ a randomly chosen conflicted variable from $csp$.VARIABLES
      $value \leftarrow$ the value $v$ for $var$ that minimizes CONFLICTS($var$, $v$, $current$, $csp$)
      set $var = value$ in $current$
   return $failure$
```

The MIN-CONFLICTS algorithm for solving CSPs by local search. The initial state may be chosen randomly or by a greedy assignment process that chooses a minimal-conflict value for each variable in turn. The CONFLICTS function counts the number of constraints violated by a particular value, given the rest of the current assignment.

Independent Subproblems

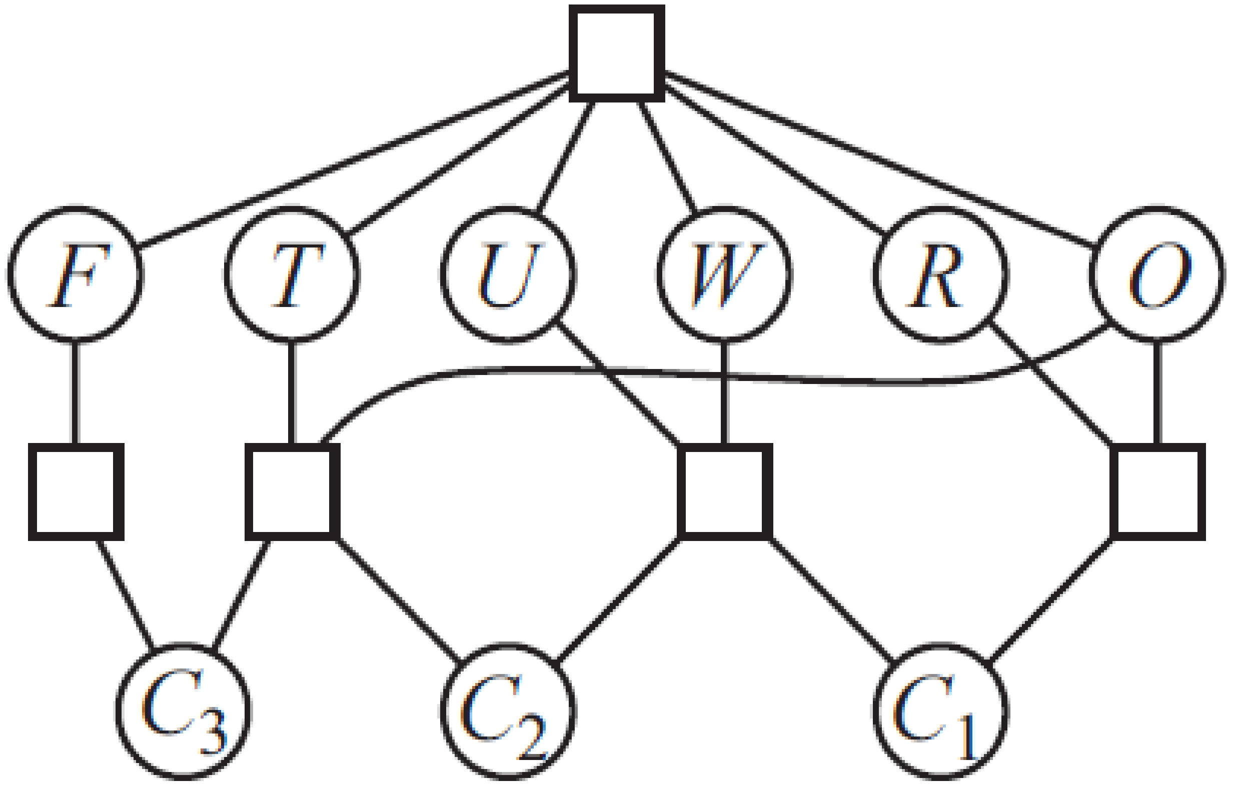
- Separate solutions then combined together
- To solve a tree-structured CSP, first pick any variable to be the root of the tree, and choose an ordering of the variables such that each variable appears after its parent in the tree – **Topological Sort**



(a) The constraint graph of a tree-structured CSP. (b) A linear ordering of the variables consistent with the tree with A as the root. This is known as a **topological sort** of the variables.

Cryptarithmetic Problem

$$\begin{array}{r} T\ W\ O \\ +\ T\ W\ O \\ \hline F\ O\ U\ R \end{array}$$


F
T
U
W
R
O
C_3
C_2
C_1