CONSTRAINT PROPAGATION: INFERENCE IN CSPS

$C_2 = 0$

$D_W = \{\cancel{0}, 3, \cancel{5}, \cancel{7}, \cancel{8}, \cancel{9}\}$

$X = \{F, T, O, W, U, R\}$

$D = \{0, 1, \ldots, 9\}$

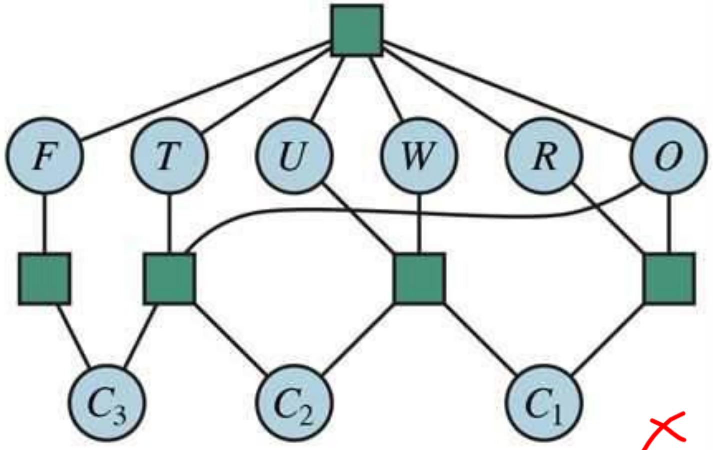
CRYPTARITHMETIC PUZZLES

$C_j \rightarrow C_1 = \text{AllDiff}$

$$
\begin{array}{cccc}
 & T & W & O \\
+ & T & W & O \\
\hline
\boxed{F} & O & U & R
\end{array}
$$

$D_F = \{1\}$

(a)



(b)

$$O + O = R + 10 \cdot C_1$$
$$C_1 + W + W = U + 10 \cdot C_2$$
$$\boxed{C_2 + T + T = O + 10 \cdot C_3}$$
$$C_3 = F,$$

× $T = 5 \Rightarrow O = \text{'}0\text{'} \Rightarrow R = \text{'}0\text{'}$

$T = 6 \Rightarrow O = 2 \Rightarrow R = 4$

$W = 3 \Rightarrow U = 6$

$F = 1$

$D_T = \{\cancel{5}, 6, 7, 8, 9\}$

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

$F = 1$

$T = 7 \Rightarrow O = 4 \Rightarrow R = 8$

$W = 3 \Rightarrow U = 6$

$\underline{\underline{C_2 = 1}}$

INFERENCE IN CSPS

- It can generate successors by choosing a new variable assignment.
- **Constraint propagation:** using the constraints to reduce the number of legal values for a variable, which in turn can reduce the legal values for another variable, and so on.

- Local Consistency:
  - Node consistency
  - Arc consistency
  - Path consistency
  - K-consistency

NODE CONSISTENCY

- If all the values in the variable's domain satisfy the variable's unary constraints.

$$SA \neq \text{Green}$$

$$D_{SA} = \{\text{Red, Blue, Green}\} \text{ -> } D_{SA} = \{\text{Red, Blue}\}$$

- A graph is node-consistent if every variable in the graph is node-consistent.

ARC CONSISTENCY

- A variable in a CSP is arc-consistent if every value in its domain satisfies the variable's binary constraints.
- X_i is arc-consistent with respect to another variable X_j if for every value in the current domain D_i there is some value in the domain D_j that satisfies the binary constraint on the arc (X_i, X_j).

$\underline{X_i} \rightarrow \underline{X_j}$

Tail Head

$X_j \rightarrow X_i$

ARC CONSISTENCY

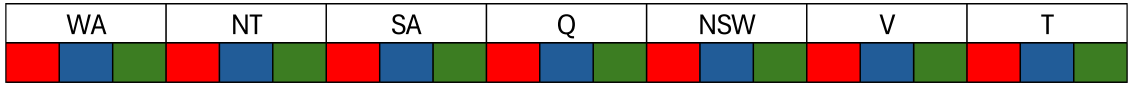
- Constraint: $Y = X^2$
- $D_X = \{0,1,2,3,4,5,6,7,8,9\}$
- $D_Y = \{0,1,2,3,4,5,6,7,8,9\}$

1. X -> Y $D_X = \{0,1,2,3\}$ $D_Y = \{0,1,2,3,4,5,6,7,8,9\}$
2. Y-> X $D_X = \{0,1,2,3\}$ $D_Y = \{0,1,4,9\}$

Modified Constraint with domain: $\langle (X,Y), \{(0,0),(1,1),(2,4),(3,9)\} \rangle$

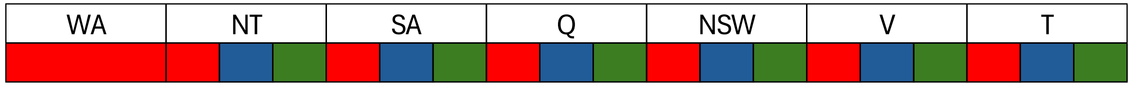
We are given the task of coloring each region either red, green, or blue in such a way that no two neighboring regions have the same color.

Initial Assignment = ϕ

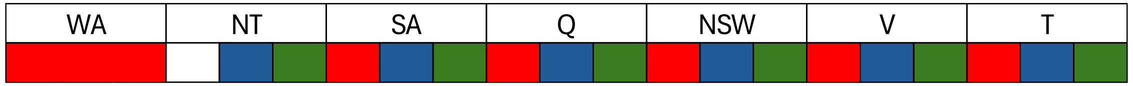


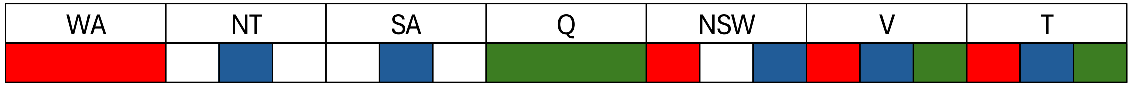
Arc Consistency does not change domain of any Variable

Initial Assignment: {WA = RED}



NT Arc Consistency with WA





1. V Arc Consistency with NSW ?

2. NSW Arc Consistency with SA ?

3. V Arc Consistency with NSW ?

AC3 ALGORITHM (ARC CONSISTENCY)

function AC-3(*csp*) **returns** false if an inconsistency is found and true otherwise
  queue ← a queue of arcs, initially all the arcs in *csp*

  while *queue* is not empty **do**
    $(X_i, X_j) \leftarrow$ POP(*queue*)
    if REVISE(*csp*, X_i, X_j) **then**
      if size of $D_i = 0$ **then return** *false*
      for each X_k **in** X_i.NEIGHBORS - $\{X_j\}$ **do**
        add (X_k, X_i) to *queue*
  return *true*

function REVISE(*csp*, X_i, X_j) **returns** true iff we revise the domain of X_i
  revised ← *false*
  for each x **in** D_i **do**
    if no value y in D_j allows (x,y) to satisfy the constraint between X_i and X_j **then**
      delete x from D_i
      revised ← *true*
  return *revised*

LIMITATIONS OF ARC CONSISTENCY

WA			NT			SA		

Checks Consistency between two Variables only!