Constraint Satisfaction Problem (CSP)

Course Title: Computational Intelligence

Course Code: CSC401A

Vaishali R Kulkarni

Department of Computer Science and Engineering
Faculty of Engineering and Technology
M. S. Ramaiah University of Applied Sciences
Email: *vaishali.c*s.et@msruas.ac.in

Constraint Satisfaction Problem (CSP)

- The **constraint** is the collection of all the restrictions and regulations that **are** imposed on the agent while solving the **problem**. The Agent cannot violate or avoid these restrictions while performing any action

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

- X is a set of variables, {X1, . . . ,Xn}.
- D is a set of domains, {D1, . . . ,Dn}, one for each variable.
- C is a set of constraints that specify allowable combinations of values.
- Each domain Di consists of a set of allowable values, {v1, . . . , vk} for variable Xi
- Each constraint Ci consists of a pair (scope, relation) , where scope is a tuple of variables that participate in the constraint and relation is a relation that defines the values that those variables can take on.

Constraint Satisfaction Problem (CSP)

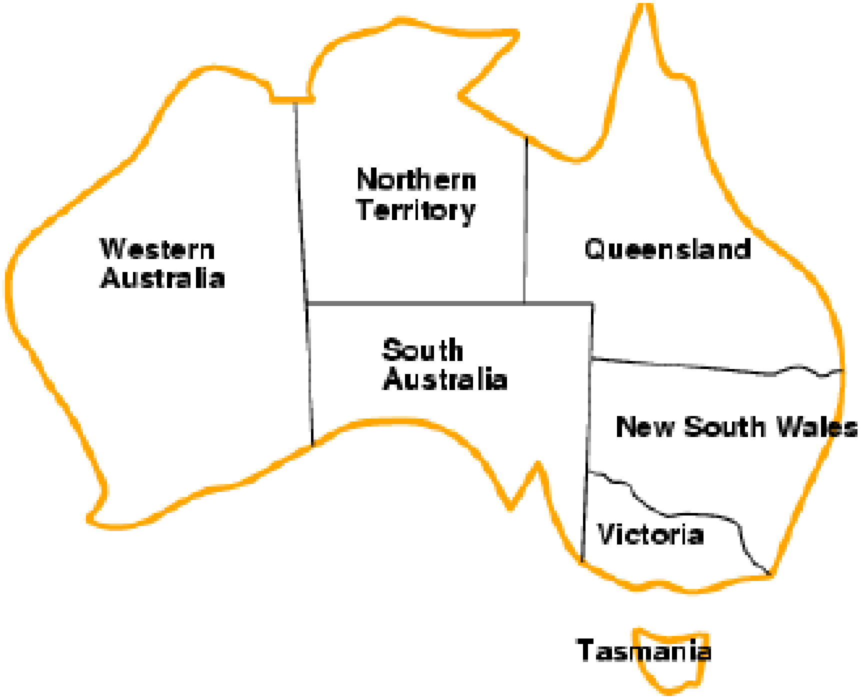
- For example, if X1 and X2 both have the domain {A,B}, then the constraint saying the two variables must have different values can be written as

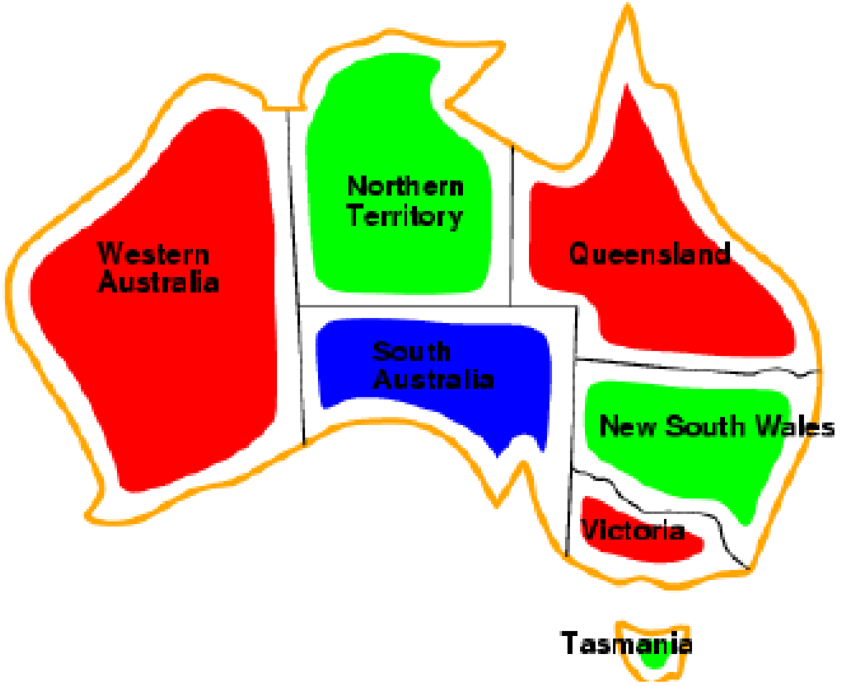
  $(X1, X2) \neq (A,B)$

- Some of the popular CSP problems include Sudoku, Cryptarithmetic, crosswords, n-Queen, etc. To solve a CSP, design the variable, domain and constraints set. Then, look for an optimal solution. The optimal solution should satisfy all constraints.
- The most used techniques are variants of backtracking, constraint propagation, and local search. The current research involves other technologies such as linear programming. Backtracking is a recursive algorithm.

Map Colouring

- **Variables:*WA, NT, Q, NSW, V, SA, T***
- Domains:Di= {red,green,blue}
- **Constraints: adjacent regions must have different colors**
- •e.g., WA ≠NT
- —So (WA,NT) must be in {(red,green),(red,blue),(green,red), …}



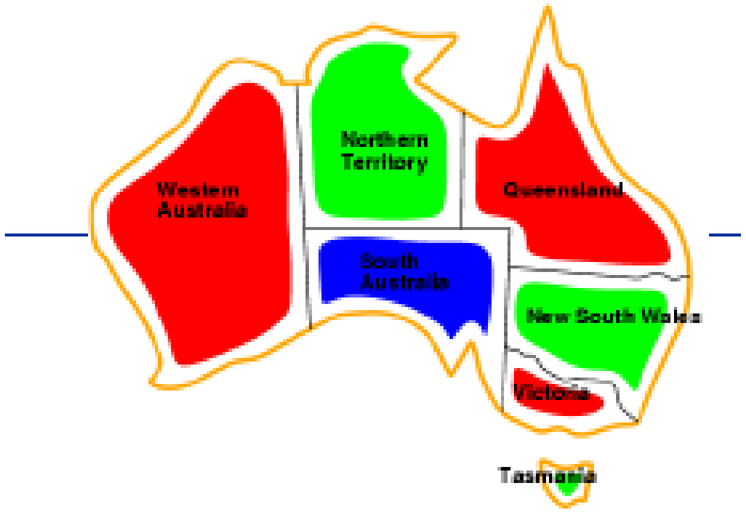


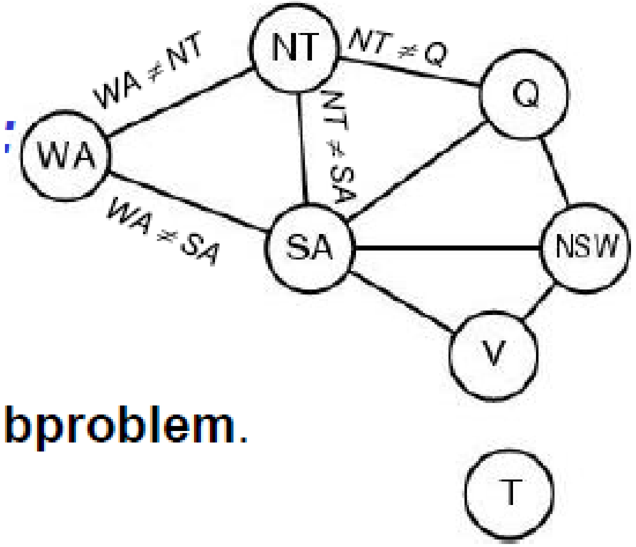
CSP Representations

- Constraint graph: •*nodes* are variables and •*arcs* are constraints
- Standard representation pattern: •variables with values
- *Constraint graph* simplifies search.

  e.g. Tasmania is an independent subproblem.
- *This problem: A binary CSP:*

  each constraint relates two variables





bproblem.

Varieties of CSPs

- Discrete variables- finite domains:

—*n*variables, domain size *d* →*O(dn)* complete assignments

—e.g., Boolean CSPs, includes Boolean satisfiability(NP-complete)

—Line Drawing Interpretation

- infinite domains: —integers, strings, etc.

—e.g., job scheduling, variables are start/end days for each job

—need a constraint language, e.g., *StartJob1+ 5 ≤ StartJob3*

- Continuous variables

e.g., start/end times for Hubble Space Telescope observations

linear constraints solvable in polynomial time by linear programming

Constraints

- Unary constraints involve a single variable,

e.g., SA ≠green

- Binary constraints involve pairs of variables,

e.g., SA ≠WA

- Higher-order constraints involve 3 or more variables

e.g., crypt-arithmetic column constraints

- Preference (soft constraints) e.g. redis better than green can be represented by a cost for each variable assignment
- Constrained optimization problems.

Constraints

- CSP can easily be expressed as a search problem
- Initial State: the empty assignment {}.
- Successor function: Assign value to any unassigned variable provided that there is not a constraint conflict.
- Goal test: the current assignment is complete.
- Path cost: a constant cost for every step.
- Solution is always found at depth n, for n variables
- Hence Depth First Search can be used

A **unary constraint** is a constraint on a single variable (e.g., $X \neq 4$). A **binary constraint** is a constraint over a pair of variables (e.g., $X \neq Y$). In general, a *k*-ary constraint has a scope of size k.

- Example: Suppose the delivery robot must carry out a number of delivery activities, a, b, c, d, and e. Suppose that each activity happens at any of times 1, 2, 3, or 4. Let A be the variable representing the time that activity a will occur, and similarly for the other activities. The variable domains, which represent possible times for each of the deliveries, are

dom(A)={1,2,3,4},dom(B)={1,2,3,4},dom(C)={1,2,3,4},

dom(D)={1,2,3,4},dom(E)={1,2,3,4}.

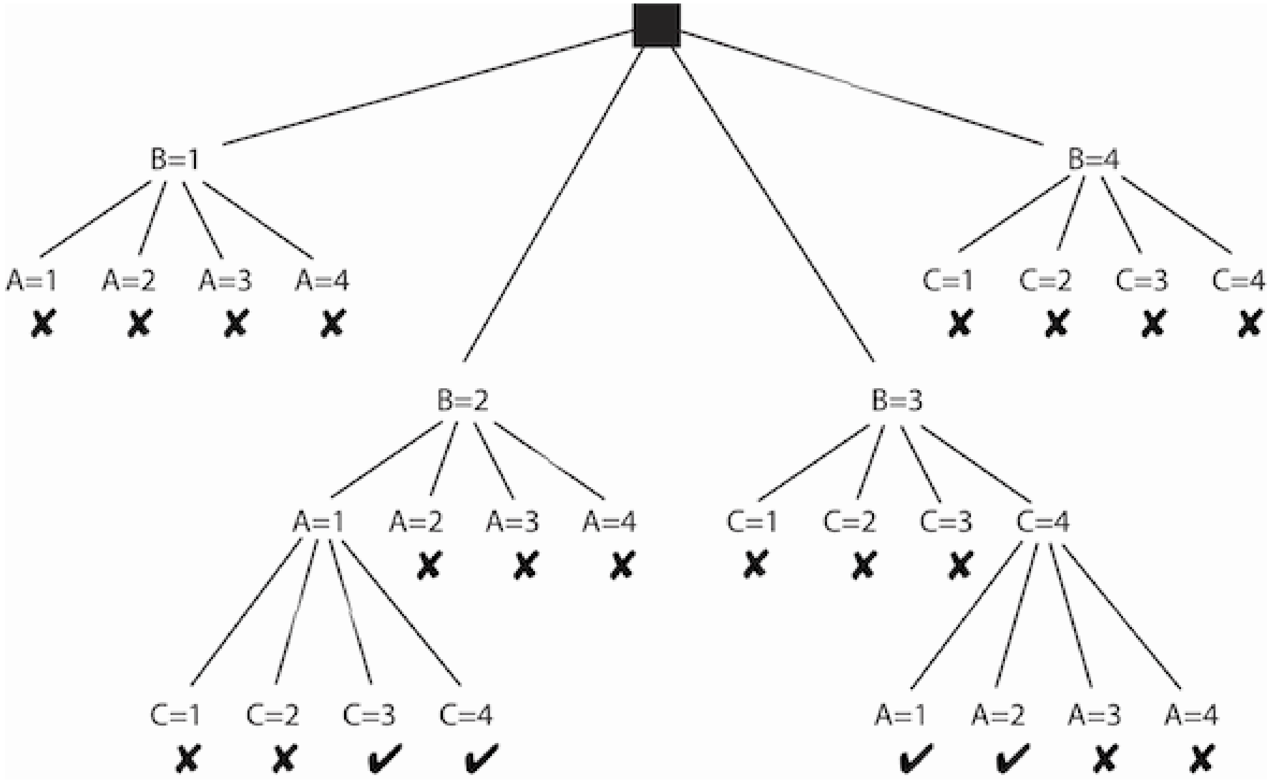
- Suppose the following constraints must be satisfied:

{(B≠3), (C≠2), (A≠B), (B≠C), (C<D), (A=D),

(E<A), (E<B), (E<C), (E<D), (B≠D)}

- The aim is to find a model, an assignment of a value to each variable, such that all the constraints are satisfied.

Solving CSPs Using Search:



- Suppose you have a CSP with the variables *A*, *B*, and *C*, each with domain *{1,2,3,4}*. Suppose the constraints *A*<*B* and *B*<*C*.
- This CSP has four solutions. The leftmost one is *A=1*, *B=2*, *C=3*. The size of the search tree, and thus the efficiency of the algorithm, depends on which variable is selected at each time. A static ordering, such as always splitting on *A* then *B* then *C*, is less efficient than the dynamic ordering used here. The set of answers is the same regardless of the variable ordering.