ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

Artificial Intelligence

Lecturer 8 – Constraint Satisfaction Problems

School of Information and Communication Technology - HUST

Constraints Satisfaction Problems (CSPs)

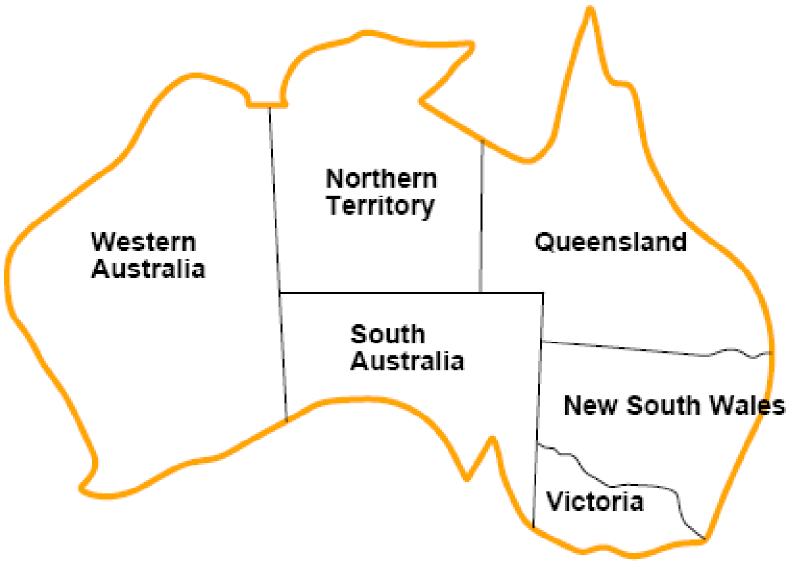
- CSPs example
- Backtracking search
- Problem structure
- Local search for CSPs

CSP

- Standard search problems
  - State is a “black-box”
    - Any data structure that implements initial states, goal states, successor function
- CSPs
  - State is composed of variables X_i with value in domain D_i
  - Goal test is a set of constraints over variables

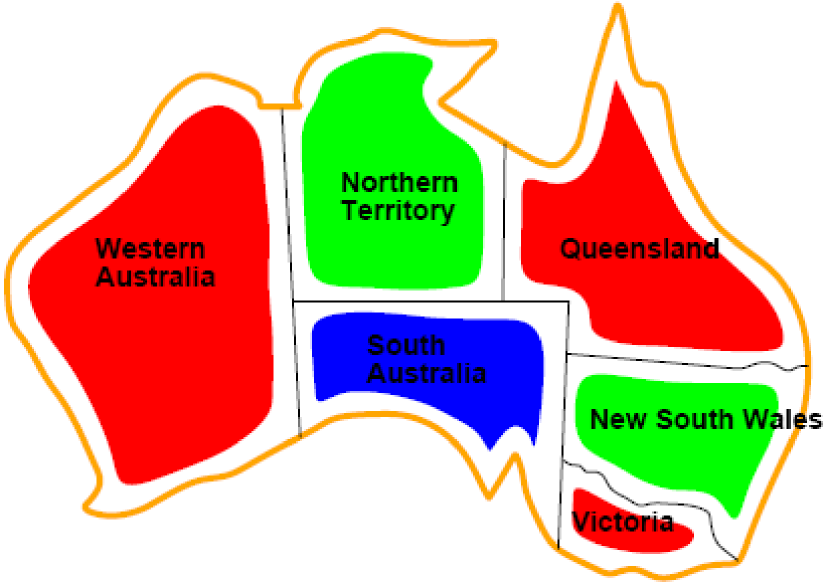
Example: Map Coloring

- Variables
  - WA, NT, Q, NSW, V , SA
- Domain
  - D_i = {red, green, blue}
- Constraint
  - Neighbor regions must have different colors
    - WA /= NT
    - WA /= SA
    - NT /= SA
    - …



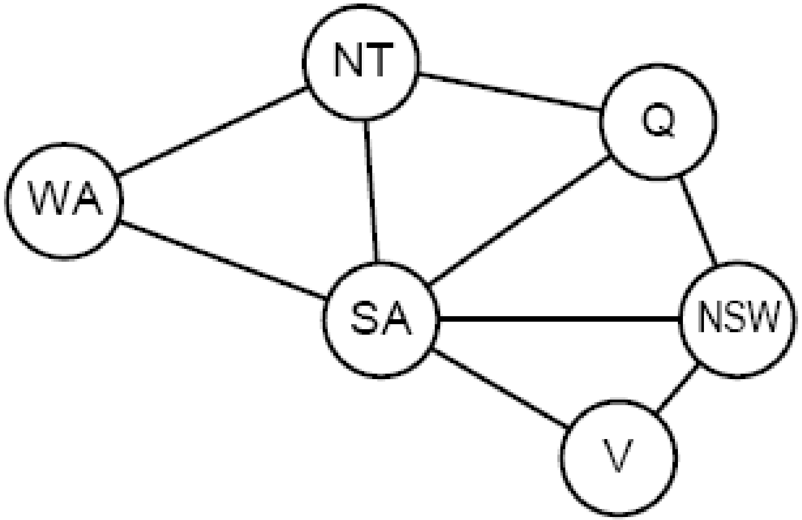
Example: Map Coloring

- Solution is an assignment of variables satisfying all constraints
  - WA=red, and
  - NT=green, and
  - Q=red, and
  - NSW=green, and
  - V=red, and
  - SA=blue



Constraint Graph

- Binary CSPs
  - Each constraint relates at most two variables
- Constraint graph
  - Node is variable
  - Edge is constraint



Varieties of CSPs

- Discrete variables
  - Finite domain, e.g, SAT Solving
  - Infinite domain, e.g., work scheduling
    - Variables is start/end of working day
    - Constraint laguage, e.g., $StartJob_1 + 5 <= StartJob_3$
    - Linear constraints are decidable, non-linear constraints are undecidable
- Continuous variables
  - e.g., start/end time of observing the universe using Hubble telescope
  - Linear constraints are solvable using Linear Programming

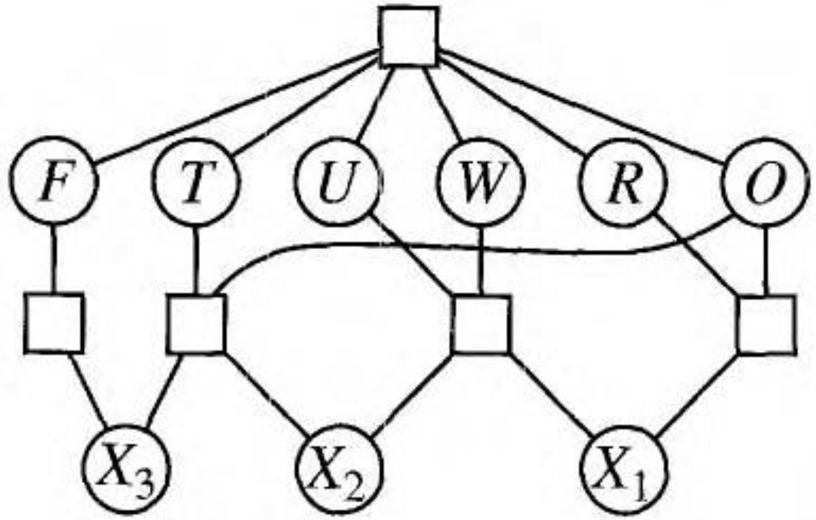
Varieties of Constraints

- Single-variable constraints
  - e.g., SA /= green
- Binary constraints
  - e.g., SA /= WA
- Multi-variable constraints
  - Relate at least 3 variables
- Soft constraints
  - Priority, e.g., red better than green
  - Cost function over variables

Example: Cryptarimetic

- Variables
  - F,T,O,U,R,W, X_1,X_2,X_3
- Domain
  - {0,1,2,3,4,5,6,7, 8,9}
- Constraints
  - Alldiff(F,T,O,U,R,W)
  - $O+O = R+10*X_1$
  - $X_1+W+W= U+10*X_2$
  - $X_2+T+T= O+10*X_3$
  - $X_3=F$

$$
\begin{array}{rr}
 & TWO \\
+ & TWO \\
\hline
 & FOUR
\end{array}
$$



Real World CSP

- Assignment
  - E.g., who teach which class
- Scheduling
  - E.g., when and where the class takes place
- Hardware design
- Spreadsheets
- Transport scheduling
- Manufacture scheduling

CSPs by Standard Search

- State
  - Defined by the values assigned so far
- Initial state
  - The empty assignment
- Successor function
  - Assign a value to a unassigned variable that does not conflict with current assignment
    - Fail if no legal assignment
- Goal test
  - All variables are assigned and no conflict

CSP by Standard Search

- Every solution appears at depth d with n variables
  - Use depth-first search
- Path is irrelevant
- Number of leaves
  - $n!d^n$
    - Two many

Backtracking Search

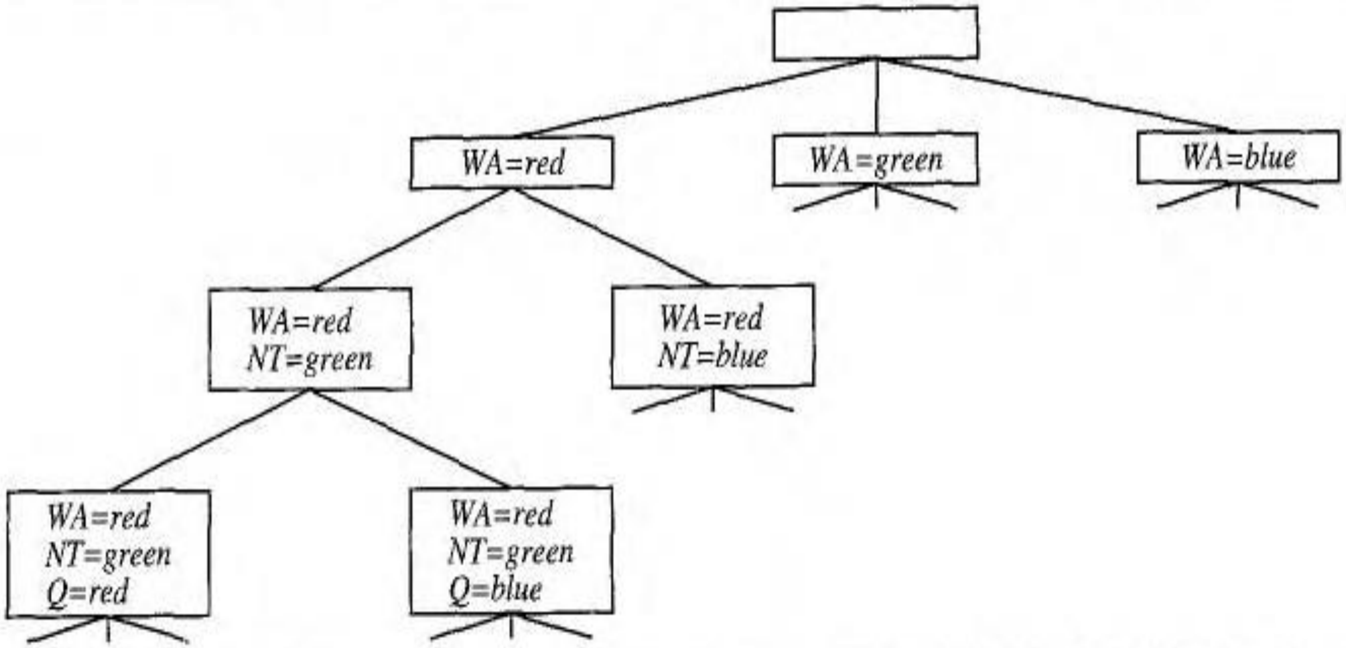
- Variable assignments are commutative, e.g.,
  - {WA=red, NT =green}
  - {NT =green, WA=red}
- Single-variable assignment
  - Only consider one variable at each node
  - d^n leaves
- Backtracking search
  - Depth-first search+ Single-variable assignment
- Backtracking search is the basic uninformed algorithm for CSPs
  - Can solve n-Queen with n = 25

Backtracking Search Algorithm

```
function BACKTRACKING-SEARCH(csp) returns solution/failure
   return RECURSIVE-BACKTRACKING({ }, csp)

function RECURSIVE-BACKTRACKING(assignment, csp) returns soln/failure
   if assignment is complete then return assignment
   var ← SELECT-UNASSIGNED-VARIABLE(VARIABLES[csp], assignment, csp)
   for each value in ORDER-DOMAIN-VALUES(var, assignment, csp) do
       if value is consistent with assignment given CONSTRAINTS[csp] then
           add {var = value} to assignment
           result ← RECURSIVE-BACKTRACKING(assignment, csp)
           if result ≠ failure then return result
           remove {var = value} from assignment
   return failure
```

Backtracking Search Algorithm



Improving Backtracking Search

- Which variable should be assigned next?
- In what order should its values be tried?
- Can we detect inevitable failure early?
- Can we take advantage of problem structure?

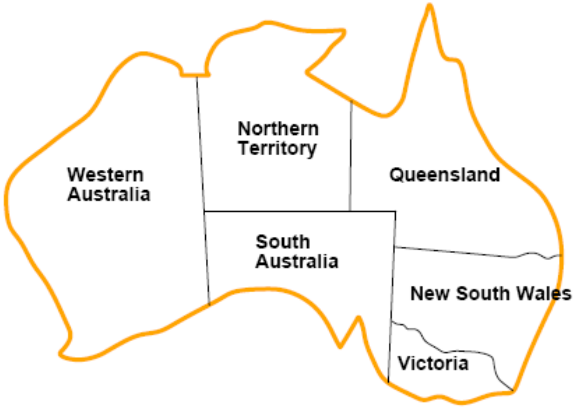
Choosing Variables

- Minimum remaining values (MRV)
  - Choose the variable with the fewest legal values
- Degree heuristic
  - Choose the variable with the most constraints on remaining variables

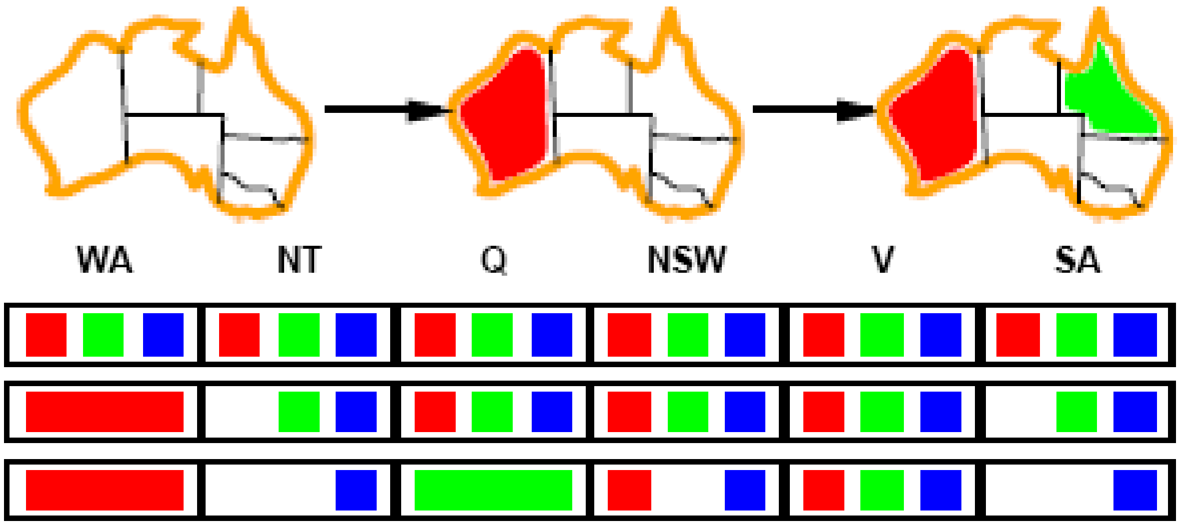
Choosing Values

- Least constraining value (LCV)
  - Choose the least constraining value
    - the one that rules out the fewest values in the remaining variables
- Keep track of remaining legal values for unassigned variables
  - Terminate search when any variable has no legal values

Forward Checking



- Constraint propagation



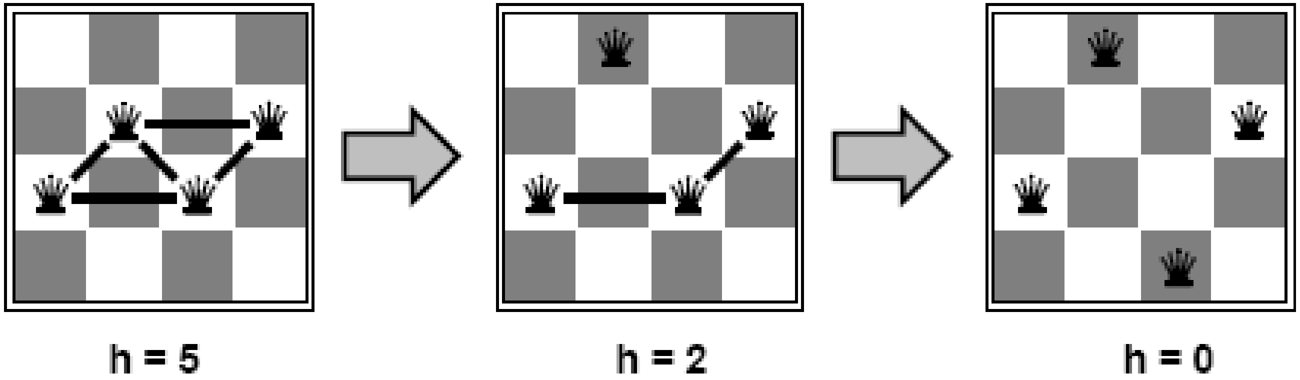
  - NT and SA cannot both be blue
- Simplest form of propagation makes each arc consistent
  - X -> Y is consistent iff for each value x of X there is some allowed value y for Y

Iterative Algorithms for CSPs

- Hill-climbing, Simulated Annealing can be used for CSPs
  - Complete state, e.g., all variables are assigned at each node
- Allow states with unsatisfiable constraints
- Operators reassign variables
- Variable selection
  - Random
- Value selection by min-conflicts heuristic
  - Choose value that violates the fewest constraints
    - i.e., hill climbing with h(n) = total number of violated constraints

Example: 4-Queens

- State: 4 queens in four columns (4*4 = 256 states)
- Operators: move queen in column
- Goal test: no attacks
- Evaluation: h(n) = number of attacks



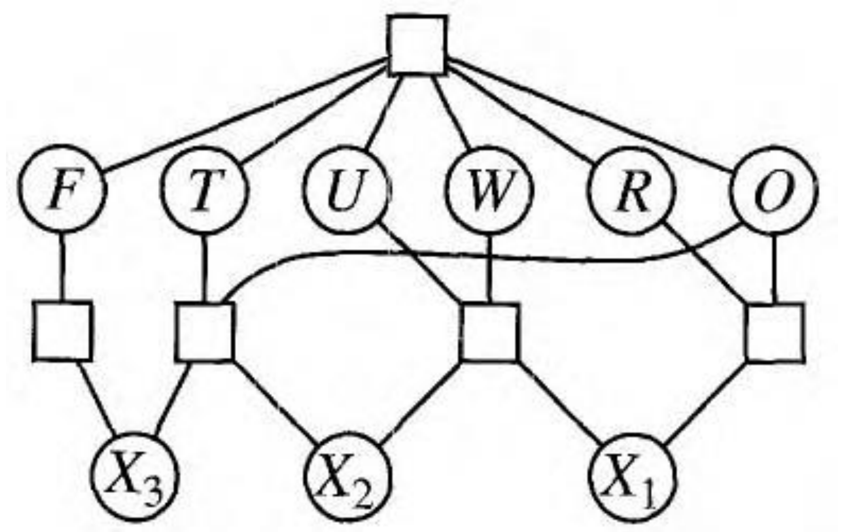
Summary

- CSPs are a special kind of problem:
  - states defined by values of a fixed set of variables
  - goal test defined by constraints on variable values
- Backtracking = depth-first search with one variable assigned per node
- Variable ordering and value selection heuristics help significantly
- Forward checking prevents assignments that guarantee later failure
- Constraint propagation (e.g., arc consistency) does additional work to constrain values and detect inconsistencies
- The CSPs representation allows analysis of problem structure
- Tree-structured CSPs can be solved in linear time
- Iterative min-conflicts is usually effective in practice

Exercice

- Solve the following cryptarithmetic problem by combining the heuristics
  - Constraint Propagation
  - Minimum Remaining Values
  - Least Constraining Values

$$\begin{array}{r} \text{TWO} \\ +\ \ \text{TWO} \\ \hline \text{FOUR} \end{array}$$



Exercice

- $O+O = R+10*X_1$
- $X_1+W+W= U+10*X_2$
- $X_2+T+T= O+10*X_3$
- $X_3=F$

1. Choose X_3: domain {0,1}
2. Choose $X_3=1$: use constraint propagation F/=0
3. F = 1
4. Choose X_2: X_1 and X_2 have the same remaing values
5. Choose $X_2=0$
6. Choose X_1: X_1 has Minimum remaining values (MRV)
7. Choose $X_1=0$
8. Choose O: O must be even, less than 5 and therefore has MRV (T+T=O dư 1 và O+O=R+10*0)
9. Choose O=4
10. R=8
11. T=7
12. Choose U: U must be even, less than 9
13. U=6: constraint propagation
14. W=3