Artificial Intelligence & Machine Learning

Bodhayan Roy

Department of Mathematics,
Indian Institute of Technology Kharagpur

Lecture 8

Constraint satisfaction problems

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

- X is a set of variables, $\{X_1, ..., X_n\}$.
- D is a set of domains, $\{D_1, ..., D_n\}$, one for each variable.
- C is a set of constraints that specify allowable combinations of values.

Constraint satisfaction problem

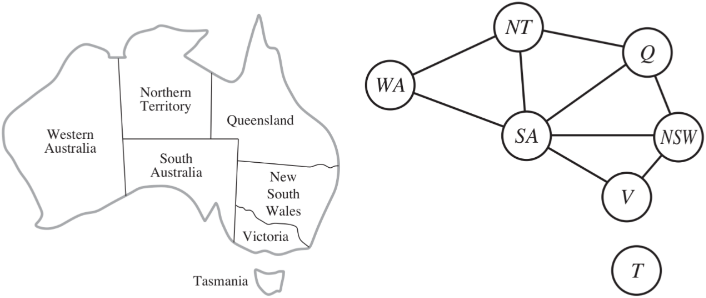
Each domain D_i consists of a set of allowable values, $\{v_1, ..., v_k\}$ for variable X_i . Each constraint C_i consists of a pair (scope, rel), where scope is a tuple of variables that participate in the constraint and rel is a relation that defines the values that those variables can take on. A relation can be represented as an explicit list of all tuples of values that satisfy the constraint, or as an abstract relation that supports two operations: testing if a tuple is a member of the relation and enumerating the members of the relation.

Constraint satisfaction problems

To solve a CSP, we need to define a state space and the notion of a solution. Each state in a CSP is defined by an assignment of values to some or all of the variables, $\{X_i = v_i, X_j = v_j, ...\}$. An assignment that does not violate any constraints is called a consistent or legal assignment. A complete assignment is one in which every variable is assigned, and a solution to a CSP is a consistent, complete assignment. A partial assignment is one that assigns values to only some of the variables.

Example: Map coloring

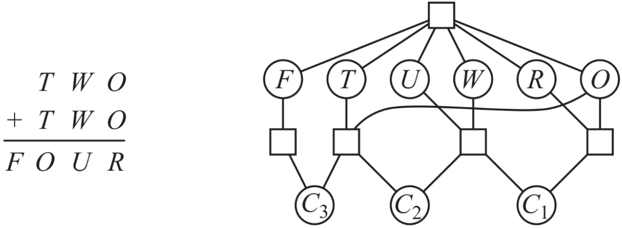
Color each region either red, green, or blue in such a way that no neighboring regions have the same color.



(a) (b)

Example: cryptarithmetic puzzles

Color each region either red, green, or blue in such a way that no neighboring regions have the same color.



(a) (b)

Constraint satisfaction problems

- A single variable (corresponding to a node in the CSP network) is node-consistent if all the values in the variable's domain satisfy the variable's unary constraints.
- A variable in a CSP is arc-consistent if every value in its domain satisfies the variable's binary constraints.
- A network is arc-consistent if every variable is arc consistent with every other variable.

Arc-consistency algorithm

```
function AC-3(csp) returns false if an inconsistency is found and true otherwise
  inputs: csp, a binary CSP with components (X, D, C)
  local variables: queue, a queue of arcs, initially all the arcs in csp

  while queue is not empty do
    (X_i, X_j) ← REMOVE-FIRST(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
```

Constraint satisfaction problems

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 Xm that satisfies the constraints on $\{X_i, X_m\}$ and $\{X_m, X_j\}$. This is called path consistency because one can think of it as looking at a path from X_i to X_j with X_m in the middle.

Constraint satisfaction problems

A CSP is k-consistent if, 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. 1-consistency says that, given the empty set, we can make any set of one variable consistent: this is what we called node consistency. 2-consistency is the same as arc consistency. For binary constraint networks, 3-consistency is the same as path consistency.

Constraint satisfaction problems

A CSP is strongly k-consistent if it is k-consistent and is also (k - 1)-consistent, (k - 2)-consistent, . . . all the way down to 1-consistent.

Backtracking algorithm for CSPs

```
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
```