Constraint Satisfaction Problems (CSPs)

CPS 570

Ron Parr

CSPs

- What is a CSP?
- One view: Search with special goal criteria
- CSP definition (general):
  - Variables $X_1,\ldots,X_n$
  - Variable X_i has domain D_i
  - Constraints $C_1,\ldots,C_m$
  - Solution: Each variable gets a value from its domain such that no constraints violated
- CSP examples…
  - http://www.csplib.org/

Other CSP Examples

- Satisfying curriculum/major requirements
- Sudoku
- Seating arrangements at a party
- LSAT Questions:
  http://www.lsac.org/JD/pdfs/SamplePTJune.pdf

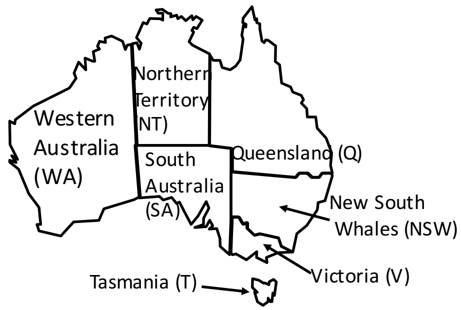
A Restricted View

- Variables $X_1,\ldots,X_n$
- A binary constraint, lists permitted assignments to pairs of variables
- A binary constraint between binary variables is a table of size 4, listing legal assignments for all 4 combinations.
- A k-ary constraint lists legal assignments to k variables at a time.
- How large is a k-ary constraint for binary variables?

Note: More expressive languages are often used.

CSP Example

Graph coloring:

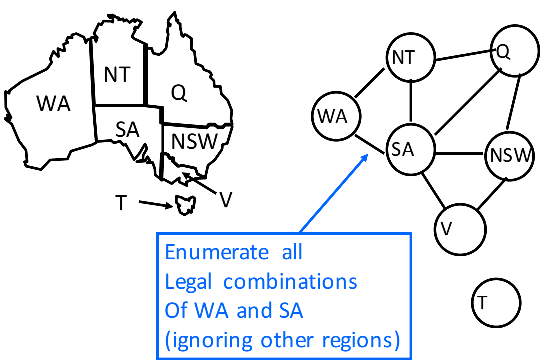


Problem: Assign Red, Green and Blue so that no 2 adjacent regions have the same color. (3-coloring)

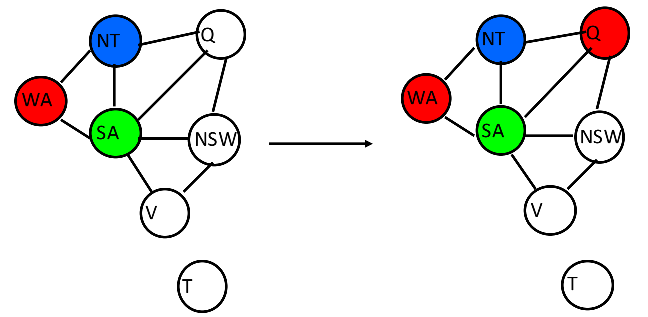
Example Contd.

- Variables: {WA, NT, Q, SA, NSW, V, T}
- Domains: {R,G,B}
- Constraints:
  For WA – NT:{(R,G), (R,B), (G,B), (G,R), (B,R), (B,G)}
- We have a table for each adjacent pair
- Are our constraints binary?
- Can every CSP be viewed as a graph problem?

Constraint Graph



CSPs as Search



Nodes: Partial Assignments

Actions: Make Assignments

Backtracking

- Backtracking is the most obvious (and widely used) method for solving CSPs:
  - Search forward by assigning values to variables
  - If stuck, undo the most recent assignment and try again
  - Repeat until success or all combinations tried
- Embellishments
  - Methods for picking next variable to assign (e.g. most constrained)
  - Backjumping

NP-Completeness of CSPs

- Are CSPs in NP?
- Are they NP-hard?

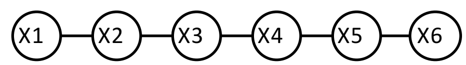
- CSPs and graph coloring are equivalent
  - Convert any graph coloring problem to CSP
  - Convert any CSP to graph coloring
- Known: Graph coloring is NP-complete
- CSPs are NP-complete
- End of the story or just the beginning?

Constraint Graphs

- Constraint graphs are important because they capture the structural relationships between the variables

- IMPORTANT CONCEPT:
  Not all instances of a hard problem class are hard
  - Structural features give insight into hardness
  - Example: Planar graphs are known to be 4-colorable
  - Group problems within class by structural features
  - New measure of problem complexity

Linear Constraint Structures



Are these easy or hard?

Properties of Chains

Theorem: Any chain of length n can be 2-colored

Proof: Induction on n.

Base: Chains of length 1 can be 2-colored.

I.H. Chains of length i can be 2-colored.

I.S. Extending an i step chain by 1 new arc consistent link produces an i+1 link chain that can be 2-colored.

Proof of I.S.: 2-color the length i chain, then color the new link with a color different from the node to which it is connected

Properties of Trees

Theorem: k-colorability of trees can be verified in polynomial time.

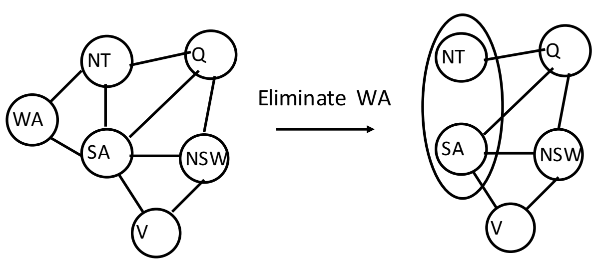
Proof: Generalize the chain case…

Corollary: Hardness of CSPs with constraint trees is

Polynomial!

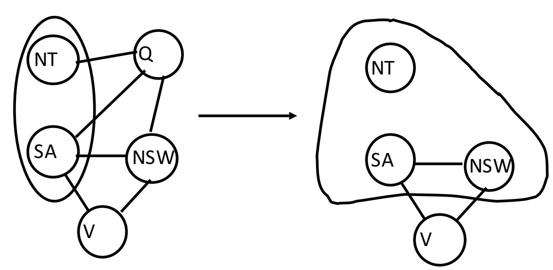
Cool fact: We now have a graph-based test for separating out some of the hard problems from the easy ones.

Variable Elimination



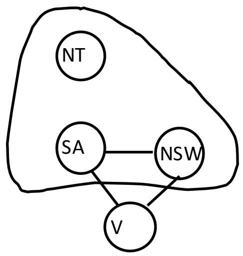
Domain(NT,SA) = {(blue, green), (blue, red), (green, blue), (green, red), (red, blue), (red, green)}

Eliminate Q



Domain(NT,SA,NSW) = {(blue, green, blue), (blue, red, blue), (red, blue, red), (red, green, red), (green, blue, green), (green, red, green)}

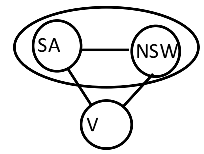
Simplify



Domain(SA, NSW) =
{(blue, green), (blue, red),
(green, blue), (green, red),
(red, blue), (red, green)}

Domain(NT,SA,NSW) = {(blue, green, blue), (blue, red, blue),
(red, blue, red), (red, green, red), (green, blue, green),
(green, red, green)}

Finish



Domain(SA, NSW) =
{(blue, green), (blue, red),
(green, blue), (green, red),
(red, blue), (red, green)}

Can identify all settings of SA, V, NSW for which there is guaranteed to be a consistent setting of the remaining variables.

Q: How do we get the settings of the other variables?

Variable Elimination

Var_elim_CSP_solve (vars, constraints)
Q = queue of all variables
i = length(vars)+1
While not(empty(Q))
  X = pop(Q)
  Xi = merge(X, neighbors(X))
  Simplify Xi (remove variables w/o external connections)
  remove_from_Q(Q, neighbors(X))
  add_to_Q(Q, Xi)
  i=i+1

Note: Merge operation can be tricky to implement, depending upon constraint language.

Variable Elimination Issues

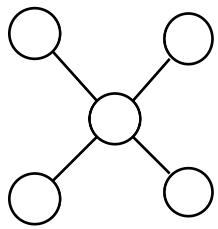
- How expensive is this?

  Exponential in size of largest merged variable set.

- Is it sensitive to elimination ordering?

  Yes!

Variable Elimination Ordering



Is it better to start at the edges and work in, or at the center and work out?

Edges!

Variable Elimination Facts

- You can figure out the cost of a particular elimination ordering without actually constructing the tables
- Finding optimal elimination ordering is NP hard
- Good heuristics for finding near optimal orderings
- Another structural complexity measure
- Investment in finding good ordering can be amortized

CSP Summary

- CSPs are a specialized language for describing certain types of decision problems
- In general, CSPs are NP hard – no general, fast solutions on the horizon
- In some cases, we can use structural measures of complexity to figure out which ones are really hard