CSPs I
CPS 271
Ron Parr

Digression: NP-hardness
- Many problems in AI are NP-hard (or worse)
- What does this mean?
- These are some of the hardest problems in CS
- Identifying a problem as NP hard means:
  - You probably shouldn’t waste time trying to find a polynomial time solution
  - If you find a polynomial time solution, either
    - You have a bug
    - You need to find a place on your shelf for your Turing award
- NP hardness is a major triumph (and failure) for computer science theory

What is the class NP?
- A class of decision problems (Yes/No)
- Solutions can be verified in polynomial time
- Examples:
  - 3SAT:
    \[(X_1 \lor \overline{X}_1 \lor X_{13}) \land (X_2 \lor X_{12} \lor X_{23}) \land \ldots\]
  - TSP (existence):
    \[\ldots\]
  - Sortedness: [1 2 3 4 5 8 7]

What is NP completeness?
- All NP complete problems can be “reduced” to each other in polynomial time
- What is a reduction?
  - Use one problem to solve another
  - A is reduced to B, if we can use B to solve A:
    \[\text{A instance} \quad \text{Poly-time transformation} \quad \text{B Solver}\]
    \[\text{poly time A solver if B is poly time}\]

Why care about NP-completeness?
- Solving any one NP-complete problem gives you the key to all others
- NP-complete problems are, in a sense, equivalent
- Insight into solving any one gives you insight into solving a vast array of problems of extraordinary practical and economic significance

Proving NP Completeness
- Want to prove problem C is NP complete
  - Show that C is in NP
  - Find known NP complete problem reducible to C
  - Are TSPs NP-complete?
    - Change into a decision problem
    - Q: Does there exist a solution with cost < k
      - Implies poly time solution to optimization problem
    - Prove that TSPs are in NP
    - Reduce known NP complete problem to TSPs
What is NP Hardness

- NP hardness is weaker than NP completeness
- \( C \) is NP hard if an NP complete problem is reducible to it
- NP completeness = NP hardness + NP membership
- Consider the problem \#SAT
  - How many satisfying assignments to:
    \[ (X_1 \lor \overline{X_1} \lor X_{11}) \land (X_2 \lor X_{12} \lor \overline{X_{21}}) \land \ldots \]
  - Is this in NP?
  - Is it NP-hard?

#SAT is NP-hard

- Theorem: \#SAT is NP hard
- Proof:
  - Reduce SAT to \#SAT

\[
\begin{align*}
\text{SAT instance} & \rightarrow \text{\#SAT solver} \\
& \rightarrow x \\
& \rightarrow \text{if } x > 0 \text{ return Y else return N}
\end{align*}
\]

False Starts

- Using exponential time in the reduction
  - OTW, you can embed the solution to the original problem in the transformation
- Forgetting to check the biconditional
  - Transformed solver must give exactly the same answers as a true solver
  - For example, transformed SAT solver must say Y iff original instance is satisfiable

NP-Completeness Summary

- NP-completeness tells us that a problem belongs to class of similar, hard problems.
- What if you find that a problem is NP hard?
  - Look for good approximations
  - Find different measures of complexity
  - Look for tractable subclasses
  - Use heuristics

Back to CSPs

- What is a CSP?
- One view: Subclass of search problems with special goal criteria
- CSP definition:
  - 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 Example

Graph coloring:

Problem: Assign colors, Red, Green and Blue so that no 2 adjacent Regions have the same color.
Example Contd.

- Variables: \{WA, NT, Q, SA, NSW, V, T\}
- Domains: \{R, G, B\}
- Constraints: not(WA(R), NT(R)), not(WA(B), NT(B)), not(WA(G), NT(G)),…
- In this case, we have 3 constraints for all adjacent pairs
- Note: Constraint language is important; here we are assuming a logical formulation
- Check the GRE!

Issues

- What are good heuristics?
  - Often good to think of this as a local search
  - Focus on choosing actions carefully, instead of pruning nodes carefully
- Can we develop heuristics that apply to the entire class of problems, not just specific instances?
- What’s the best we can hope for?

NP-Completeness of CSPs

- Are CSPs in NP?
- Solving SAT with a CSP
- SAT variables map to CSP variables
- Disjunctions map to constraints
- All constraints are implicitly conjoined
- Is graph coloring NP-complete?

Heuristics

- Pick next variable to change next:
  - Most constrained first
  - Most constraining first
  - Least constraining value first
Most Constrained Variable

Most Constraining Variable

Forced!

Causes greatest reduction in choices for other variables.