CS188 Section 4: CSPs and Propositional Logic

1 Course Scheduling

You are in charge of scheduling for computer science classes that meet Mondays, Wednesdays and Fridays. There are 5 classes that meet on these days and 3 professors who will be teaching these classes. You are constrained by the fact that each professor can only teach one class at a time.

The classes are:

1. Class 1 - Intro to Programming: meets from 8:00-9:00am
2. Class 2 - Intro to Artificial Intelligence: meets from 8:30-9:30am
3. Class 3 - Natural Language Processing: meets from 9:00-10:00am
4. Class 4 - Computer Vision: meets from 9:00-10:00am
5. Class 5 - Machine Learning: meets from 10:30-11:30am

The professors are:

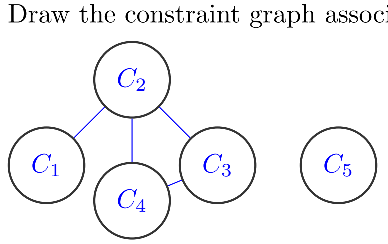
1. Professor A, who is qualified to teach Classes 1, 2, and 5.
2. Professor B, who is qualified to teach Classes 3, 4, and 5.
3. Professor C, who is qualified to teach Classes 1, 3, and 4.

1. Formulate this problem as a CSP problem in which there is one variable per class, stating the domains, and constraints. Constraints should be specified formally and precisely, but may be implicit rather than explicit.

Variables	Domains (or unary constraints)
C_1	{A, C}
C_2	{A}
C_3	{B, C}
C_4	{B, C}
C_5	{A, B}

Binary Constraints
$C_1 \neq C_2$
$C_2 \neq C_3$
$C_2 \neq C_4$
$C_3 \neq C_4$

2. Draw the constraint graph associated with your CSP.



3. Your CSP should look nearly tree-structured. Find the cutset for the constraint graph and explain why it's useful.

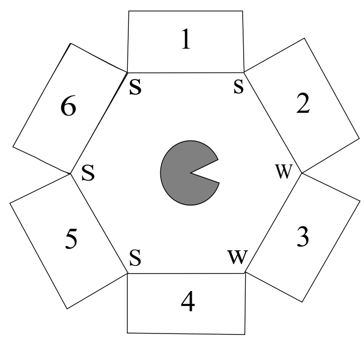
   Minimal answer: we can solve them in polynomial time. If a graph is tree structured (i.e. has no loops), then the CSP can be solved in $O(nd^2)$ time as compared to general CSPs, where worst-case time is $O(d^n)$. For tree-structured CSPs you can choose an ordering such that every node's parent precedes it in the ordering. Then after enforcing arc consistency you can greedily assign the nodes in order, starting from the root, and will find a consistent assignment without backtracking.

2 Trapped Pacman

Pacman is trapped! He is surrounded by mysterious corridors, each of which leads to either a pit (P), a ghost (G), or an exit (E). In order to escape, he needs to figure out which corridors, if any, lead to an exit and freedom, rather than the certain doom of a pit or a ghost.

The one sign of what lies behind the corridors is the wind: a pit produces a strong breeze (S) and an exit produces a weak breeze (W), while a ghost doesn't produce any breeze at all. Unfortunately, Pacman cannot measure the strength of the breeze at a specific corridor. Instead, he can stand *between* two adjacent corridors and feel the max of the two breezes. For example, if he stands between a pit and an exit he will sense a strong (S) breeze, while if he stands between an exit and a ghost, he will sense a weak (W) breeze. The measurements for all intersections are shown in the figure below.

Also, while the total number of exits might be zero, one, or more, Pacman knows that two neighboring squares will *not* both be exits.



Pacman models this problem using variables X_i for each corridor i and domains P, G, and E.

1. State the binary and/or unary constraints for this CSP (either implicitly or explicitly).

   Binary:

   $X_1 = P$ or $X_2 = P$, $X_2 = E$ or $X_3 = E$,
   $X_3 = E$ or $X_4 = E$, $X_4 = P$ or $X_5 = P$,
   $X_5 = P$ or $X_6 = P$, $X_1 = P$ or $X_6 = P$,
   $\forall i, j$ s.t. $\text{Adj}(i, j)\ \neg(X_i = E \text{ and } X_j = E)$

   Unary:

   $X_2 \neq P$,
   $X_3 \neq P$,
   $X_4 \neq P$

2. Although this is not a tree-structured CSP, we can still use the concept of arc-consistency to help us remove values from the domains of the variables. The basic idea is this: for each variable, enforce the consistency of arcs from that variable. Repeat this process until no further deletions are possible (see Figure 6.3 in the book for more details on this algorithm). In the picture below, cross out the values from the domains of the variables that will be deleted in enforcing arc consistency this way.

X_1	P		
X_2		G	E
X_3		G	E
X_4		G	E
X_5	P		
X_6	P	G	E

3. According to MRV, which variable or variables could the solver assign first?

   X_1 or X_5 (tie breaking)

4. Assume that Pacman knows that $X_6 = G$. List all the solutions of this CSP or write *none* if no solutions exist.

   (P,E,G,E,P,G)
   (P,G,E,G,P,G)

5. Now let's attempt to formulate the above problem in terms of propositional logic. Again Pacman is interested in reasoning about what lies behind each corridor. First, write down a set of proposition symbols that will be useful for solving this problem.

   Note: For essentially all of the following problems, there are multiple solutions to what variables and logical sentences you can use. For these solutions, we've written down what we believe are the most intuitive and explicit solutions, although they are not necessarily the most elegant.

   P_i for i from 1 to 6, indicating if there is a pit behind corridor i
   E_i for i from 1 to 6, indicating if there is an exit behind corridor i
   G_i for i from 1 to 6, indicating if there is a ghost behind corridor i
   W_{ij} for pairs of adjacent i,j from 1 to 6, indicating if Pacman observed a weak breeze between corridors i and j
   S_{ij} for pairs of adjacent i,j from 1 to 6, indicating if Pacman observed a strong breeze between corridors i and j
   N_{ij} for full generality, for pairs of adjacent i,j from 1 to 6, indicating if Pacman observed no breeze between corridors i and j

6. Next, given what you know about the rules describing how the world works, write down sentences that relate the proposition symbols. Feel free to write generic sentences (e.g., if you have symbols A_1, A_2, A_3, etc., you can write sentences that include A_i or A_j if you specify the allowed values of i and j).

   $\neg(E_i \wedge E_j)$ for adjacent i,j from 1 to 6
   $S_{ij} \Rightarrow P_i \vee P_j$ for adjacent i,j from 1 to 6
   $N_{ij} \Rightarrow G_i \wedge G_j$ for adjacent i,j from 1 to 6
   $W_{ij} \Rightarrow (E_i \vee E_j) \wedge \neg P_i \wedge \neg P_j$ for adjacent i,j from 1 to 6
   $P_i \Rightarrow \neg E_i \wedge \neg G_i$ for i from 1 to 6
   $E_i \Rightarrow \neg P_i \wedge \neg G_i$ for i from 1 to 6
   $G_i \Rightarrow \neg E_i \wedge \neg P_i$ for i from 1 to 6

7. Continue by writing down sentences that incorporate Pacman's observations shown above.

   $W_{12} = False$, $W_{23} = True$, $W_{34} = True$, $W_{45} = False$, $W_{56} = False$, $W_{61} = False$
   $S_{12} = True$, $S_{23} = False$, $S_{34} = False$, $S_{45} = True$, $S_{56} = True$, $S_{61} = True$
   $N_{12} = False$, $N_{23} = False$, $N_{34} = False$, $N_{45} = False$, $N_{56} = False$, $N_{61} = False$

8. Give an assignment that satisfied all of your sentences. Are there other such assignments?

   Here is one assignment (there are others):
   $P_1 = True$, $P_2 = False$, $P_3 = False$, $P_4 = False$, $P_5 = True$, $P_6 = False$

$E_1 = False$, $E_2 = True$, $E_3 = False$, $E_4 = True$, $E_5 = False$, $E_6 = False$
$G_1 = False$, $G_2 = False$, $G_3 = True$, $G_4 = False$, $G_5 = False$, $G_6 = True$
For completeness, we must also mention that the W, S, and N variables are assigned as given above.

9. Suppose you run a SAT solver on your set of sentences, and it returns the first solution if finds, or says none if there are none. Explain how you could use this SAT solver to show there are no other solutions.

   Essentially we want try to solve the problem again, but with a new constraint that we don't want to get the same solution we had before. This means we can take the first solution we got, negate it, and add that as a new sentence. Then we can run the SAT solver again incorporating this new sentence, and see if it is still able to find a solution.