1. *(A simplified example of planning using resolution & Green's method)*

We use \( \text{At}(x,y,s) \) to express that \( x \) is at \( y \) in situation \( s \); and similarly \( \text{In}(x,y,s) \) to express that \( x \) is in \( y \) in situation \( s \). Constant \( B \) will refer to Bob, \( C \) to Bob's car, \( W \) to his work location, and \( S \) is our name for a certain situation in Bob’s world. You can think of a “situation” as the state of the world at a brief moment in time; actions (represented as functions on situations) change that situation to a new one at a later moment. The ‘get-in-car’ function represents the action of Bob getting in his car, and this maps any given situation \( s \) to a new one, namely, \( \text{get-in-car}(s) \). Similarly the ‘drive-to-work’ function represents the action of Bob driving to work, which changes the starting situation to one where Bob is at work (\( W \)).

a. (6) Given:
\[
\begin{aligned}
\text{At}(B,C,S), \\
\forall s. \text{At}(B,C,s) \Rightarrow \text{In}(B,C,\text{get-in-car}(s)) \\
\forall s. \text{In}(B,C,s) \Rightarrow \text{At}(B,W,\text{drive-to-work}(s))
\end{aligned}
\]

Use resolution to prove that there is a situation in which Bob is at work.

b. (6) Use the proof in (a) to answer the question “In what situation is Bob at work?”, by attaching Cordell Green’s answer literal and carrying it through the proof, with the appropriate substitutions.

c. (3) Intuitively interpret the answer obtained, keeping in mind the explanations above of what the functions mean.

2. *(Natural Logic)*

a. (6) The simplest example of Natural Logic (NLog) inference that we saw in class was “Every dog barks” — “Every poodle makes noise”. Provide a quick resolution derivation of the conclusion *without* assuming falsity of that conclusion, i.e., a forward proof. (Though resolution isn’t complete for forward inference, it’s still pretty powerful.)

b. (6) An NLog inference not readily expressible in standard FOL is this:

\[
\begin{aligned}
\text{Governor Greg prevented a million would-be voters from casting ballots} & \vdash \text{“A governor prevented many would-be voters from voting”;} \\
& \vdash \text{“Many would-be voters did not vote”}. \\
\end{aligned}
\]

Outline the properties of “prevent”, both in terms of what it implicates and
the polarity of its complement, that enable these inferences; also mention any generalization/specialization relations that need to be used (Carefully: The polarity properties here are different from what you might guess at first sight.)

c. (3) Represent “Many voters did not vote” in Episodic Logic (EL). Do this by analogy with the examples in the slides. Include a constant for the episode of many voters not voting.

3. (Satisfiability testing)
(6) Solve the following satisfiability problem with the DPLL method, stating and justifying your steps where these are constrained by that method:
\[ \neg A \lor B \lor C \lor D, \quad C \lor \neg E \lor F, \quad \neg C \lor G, \quad \neg D \lor \neg E \lor G, \quad \neg C \lor F \lor G \lor H, \quad A \lor C \lor \neg H, \quad \neg H \]

4. (Neural net reasoning)
Recall the equation that appears in the “neural tensor network reasoning” paper by Socher et al.:

\[
g(e_1, R, e_2) = u^T.f \left( e_1^T W^{[1:k]} e_2 + V \left[ \begin{array}{c} e_1 \\ e_2 \end{array} \right] + b \right).
\]

Let’s consider the simplest possible version of this, by choosing \( d = 2 \) (each of the two input “words”, \( e_1 \) and \( e_2 \), is characterized by just 2 numerical features), and \( k = 1 \) (we’re allowing for just one multiplicative interaction between the two words; the weight tensor has just one “slice”, i.e., it’s a 2 by 2 coefficient matrix). Let’s write \( e_1^T \) as \((x, y)\) and \( e_2^T \) as \((u, v)\), and let the 1st and 2nd rows of \( W \) be \((a, b)\) and \((c, d)\).

a. (6) Notice that the order of vector-matrix-vector multiplication is ambiguous in the Socher equation. That’s because such multiplications are associative. Since not everyone may know this, you’re asked to prove it for the simple case here, given the above vector and matrix elements.

b. (4) Given the value of the vector-matrix-vector product that you worked out in (a), can we adjust the weights on the product terms (for the elements of \( e_1 \) and \( e_2 \)) to anything we want? Why or why not? What does your answer imply, intuitively, about the kinds of interactions between the two “words” that Socher et al. are allowing for?

c. (9) While it’s not obvious how we can form products of numbers with standard NN units, we can easily do this for two 0/1-valued binary digits; after all, their product is 1 if and only if both of them are 1 – i.e., product is the same as \( \text{AND} \). Show and explain a single NN unit (a simple perceptron) that computes the product (AND) of two inputs. Having done that, you’ll have no trouble showing a unit that computes \( \text{OR} \). Do that. And then there’s \( \text{NOT} \). Do that one as well.