A GIRL HAS NO NAME
Computer Science 81
Homework 7
3/7/17

1, 2, 3, 4

1. [10 points] Resolution is really a proof technique best suited for computers. But just to make sure you understand how it works, we're asking you to do one resolution proof "by hand." [Do not use Prover9 on this problem until you've succeeded!]

Consider the following argument

- Every fungus is a mushroom or a toadstool.
- Every boletus is a fungus. $[\forall x.\ (B(x) \rightarrow F(x))]$
- All toadstools are poisonous.
- No boletus is a mushroom.
- Specimen b is a boletus.
- *Therefore*: Specimen b is poisonous.

A. Convert this argument into a set of clauses that is unsatisfiable if the argument is valid.

B. In a numbered list of clauses, show how to take these clauses and derive $\perp$.

■

2. [5 points] Confirm that Prover9 can do the same proof. Specifically, convert your clauses into Prover9 “assumption” syntax (dont use the goal window), and run Prover9. Then copy and paste the resulting proof (just the numbered lines) into your answer document.

■

3. [9 points] The last page of this assignment contains Prover9 code to "solve" the famous Towers of Hanoi problem. (If youre not familiar with this puzzle, we suggest you Google it.)

There are 5 disks (numbered 1 through 5, with disk 1 being smallest and disk 5 being biggest). The disks are on three pegs, with the requirement that a larger disk cannot be stacked on top of a smaller disk. So the state of the system at any moment can be described with three (sorted) lists that say which peg contains which disks. Of course, every disk must appear on some pole. So possible states include

$\mathbf{state}([\mathbf{1},\mathbf{2},\mathbf{3},\mathbf{4},\mathbf{5}],[\,],[\,])$	% starting state, where all disks are on the first peg.
$\mathbf{state}([\,],[\,],[\mathbf{1},\mathbf{2},\mathbf{3},\mathbf{4},\mathbf{5}])$	% target state, with all disks on the last peg.
$\mathbf{state}([\mathbf{5}],[\mathbf{4},\mathbf{2}],[\mathbf{1},\mathbf{3}])$	% intermediate state, where every peg has some disks.

We can move one disk at a top, moving the top disk from one peg to another, always with the constraint that we cannot put a larger disk on top of a smaller disk. The formula **transition**($\mathbf{x},\mathbf{y}$) is defined to hold if one can get from state **x** to state **y** by moving a single disk.

Finally, the formula **reachable**($\mathbf{x}$) is defined to hold if one can get from the start state to state **x** by a sequence of transitions.

You should start by pasting the provided code (on the last page of this handout) into the "assumptions" window. Verify that Prover9 successfully complains–the assumption that one **cannot** reach the configuration where all disks are on the last peg is contradictory, and hence the target configuration **is** reachable.

3. cont. But it doesnt say why/how that configuration can be reached. Your job is to modify this code so that we can get the sequence of moves out of Prover9.

A. We can describe moves with a 2-digit number. Specifically **13** means "move a disk from peg 1 to peg 3", while **32** means "move a disk from peg 3 to peg 2".

   Change the rules defining the transition relation to make it a three-argument predicate, so that $\mathbf{transition}(\mathbf{x}, \mathbf{y}, \mathbf{why})$ if we can get from state $\mathbf{x}$ to state $\mathbf{y}$ in one move, and why is the corresponding number for that move.

   Hint: the first line will be $\mathbf{transition(state}([\mathbf{w} : \mathbf{x}], [\,], \mathbf{z}),\ \ \mathbf{state}(\mathbf{x}, [\mathbf{w}], \mathbf{z}), \mathbf{12})$.

B. You cant test the code yet, because you have defined transition to be a 3-argument relation, and the definition of reachable treats it as a 2-place relation. Lets patch this by adding an extra dummy argument **why** to the definition of run:

   $$\mathbf{reachable}(\mathbf{y}) \leftarrow \mathbf{reachable}(\mathbf{x})\ \&\ \mathbf{transition}(\mathbf{x}, \mathbf{y}, \mathbf{why}).$$

   Verify that Prover9 still finds the puzzle solvable, but doesnt explain how.

C. Finally, make reachable a 2-place relation, so that $\mathbf{reachable}(\mathbf{x}, \mathbf{why})$ is true if state $\mathbf{x}$ is reachable via **why**, a list of move numbers in *reverse* order. (You can follow the model of the peg puzzle in class).

D. Test your definition by changing the final assumption to

   $$\mathbf{\text{-}reachable(state}([\,], [\,], [\mathbf{1}, \mathbf{2}, \mathbf{3}, \mathbf{4}, \mathbf{5}]), \mathbf{whys})\ \#\ \mathbf{answer(whys)}.$$

   Verify that Prover9 can discover a list **whys** of moves that reaches the target state.

E. Paste your modified code and the resulting (reversed) list of moves into this document.

■

4. [1 easy point] Answer this quick survey when youre done:

A. How long (in hours) did you spend working on this assignment?

B. What was the most interesting thing you learned while answering these problems?

■