Example Homework Answer

Due: *never* **Points:** *some points*

Questions

This is an example of a fully worked out answer to a question asking for a proof.

In writing your answers, realize that there will be a main point or idea in what you write. It's critical you express it clearly. For example, in this answer, the main point is the need to run the Turing machines in parallel, so if one Turing machine does not terminate, you won't wait forever to start the next one.

Next, say how you deal with the main point or idea. Here, you would run the first machine some number of steps, then the second some number of steps, then the third, then the second, then the first, and so forth. This runs the machines in parallel. This is called a *diagonalization technique* (to see why, look at the picture below).

Finally, you need to make the argument rigorous. That may mean to use mathematics or notations; it may mean to reason through the main point or idea (as here).

For grading, we will look first for the main idea, then how you deal with it, and lastly with the rigor. If your argument is correct but not rigorous, you will get most of the points. If you get the main idea, then you'll get a good amount of points. So clarity of expression is important.

1. (some *points*) Prove Theorem 3.3. (*Hint:* Use a diagonalization argument to test each system as the set of protection systems is enumerated. Whenever a protection system leaks a right, add it to the list of unsafe protection systems.)

   Answer: The key observation here is that we need to run the Turing machines in parallel. Otherwise, if one machine never halts, none of the others can run, and so the list of Turing machines that enter the halting state will be incomplete, and not augmented while that Turing machine is running.

   This means we have to order the Turing machines in some way. The Gödel numbers provide a natural way of doing this ordering. And by the nature of those numbers, each Turing machine has a unique one.

   Now we make the argument rigorous.

   Represent the set of all possible systems as a set of executions of Turing machines. Each such execution has a unique Gödel number. Order the executions by their Gödel numbers (each such execution is represented by TM_i, where i is the appropriate Gödel number). Now, TM_i halts in state q_f if, and only if, the right in question leaks; that is, if, and only if, the system halts in an unsafe state. If TM_i does not halt, it may be safe (and so never halt), or it may simply not yet have reached its unsafe (halting) state. This means we cannot serially execute the systems, proceeding to TM_{i+1} when TM_i halts. If we did that, and TM_i never halted, we would never begin executing TM_{i+1} and so could not enumerate the unsafe systems with Gödel numbers greater than i. So, use a diagonizational technique. Execute the first instruction in TM_1. Execute the second instruction in TM_1. Execute the first instruction in TM_2. Execute the first instruction in TM_3. Execute the second instruction in TM_2. Execute the third instruction in TM_1. Execute the fourth instruction in TM_1. Execute the third instruction in TM_2. Continue this pattern of execution, indicated in the following picture by numbers representing the order in which the steps are executed:

TM_1	1	2	6	7	15	$\cdots$
TM_2	3	5	8	14	$\cdots$	
TM_3	4	9	13	$\cdots$		
TM_4	10	12	$\cdots$			
TM_5	11	$\cdots$				
$\cdots$	$\cdots$					

   Now, as each TM_i halts, its state is checked. If it halted in state q_f, it is added to the list of unsafe systems. Otherwise, it is ignored. The diagonalization procedure is modified to skip over the halted TM_i. Thus, all systems which halt in state q_f will be enumerated. Therefore, the set of unsafe systems is recursively enumerable.