

CS 620 – Theory of Computation – Fall 2009
Instructor: Marc Pomplun

Practice Exam Solutions

Question 1: ____ out of ____ points

Question 2: ____ out of ____ points

Question 3: ____ out of ____ points

Question 4: ____ out of ____ points

Question 5: ____ out of ____ points

Total Score:

Grade:

Question 1: True or False?

Are the following statements true, false, or is their truth value unknown? Check the appropriate box for each statement.

	true	false	unknown
a) A function f is partially computable if and only if there is a program of \mathcal{L} that computes f .	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
b) For all sets A and B , if $A \leq_m B$ and B is recursive, then A is also recursive.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
c) If a non-empty set S is r.e., then S is also the range of a primitive recursive function.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
d) There are infinitely many programs in the language \mathcal{L} that compute the function $f(x) = 2x$.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
e) If during a computation by a program \mathcal{P} in the language \mathcal{L} the same state occurs more than once, then this computation will not terminate.	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
f) For all sets A and B , if $A \cap B$ is r.e., then A and B are both r.e.	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
g) For all sets A and B , if $A \cup B$ is r.e., then A and B are both r.e.	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
h) If function $g(x)$ is partially computable and if function $h(x)$ is computable, then function $f(x) = g(h(x))$ is computable.	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
i) There is an algorithm that cannot be executed by any program of the language \mathcal{L} .	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
j) For all sets A and B , if A and B are both r.e., then $A \cap B$ is also r.e.	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Question 2: The Limping Turning Machine

Imagine a Turing machine M' that works like a normal Turing machine except that whenever it moves to the left, it moves by two squares instead of one. Its movements to the right are by one square as usual. Give a detailed proof of the fact that M' can compute all functions that normal Turing machines can compute.

Hint: Think of the simulation of Turing machines with quintuple Turing machines that we discussed in class.

The basic idea is to show that whatever Turing machines can do, the machines of type M' can do as well. This would be achieved by devising a scheme for translating any given Turing machine into an equivalent machine of the new type. In other words, we want to show that we can simulate any given Turing machine with a machine of type M' .

To do this for a given Turing machine M , we take all quadruples in M of the types $q_i s_j s_k q_l$ and $q_i s_j R q_l$ and put them into our new machine M' without changing them at all. For each quadruple in M of the form $q_i s_j L q_l$, we add the quadruple $q_i s_j L^2 q_{k+1}$ to M' , where L^2 stands for the double-leap to the left. Then we add new states q_{k+1}, \dots, q_{2k} to M' , and also the quadruples $q_{k+i} s_j R q_i$ for all $i = 1, \dots, k, j = 1, \dots, n$, where n is the number of symbols in the alphabet.

Then the machine M' will perform a computation equivalent to the one performed by M , and since we can do this for any Turing machine M , we have proven that the new type of machine is at least as computationally powerful as Turing machines.

Question 3: Programmer's Proof

Show that if two sets A and B are both recursive, then $A - B$ is also recursive. In your proof, use a program in the language \mathcal{L} (possibly containing macros).

Since A and B are recursive, there are computable predicates P_A and P_B such that:

$$A = \{ x \in N \mid P_A(x) \}$$

$$B = \{ x \in N \mid P_B(x) \}$$

For our proof we have to show that there is a computable predicate P_C so that

$$A - B = \{ x \in N \mid P_{A-B}(x) \}$$

We show this by writing a program that computes $P_{A-B}(x)$:

IF $\sim P_A(X)$ GOTO E

IF $P_B(X)$ GOTO E

Y \leftarrow Y + 1

Question 4: A Useful Program

Describe in detail what the following program does. In which case would such a program be especially useful? Why?

```
[A]  IF STP(1)(X, p, T) GOTO C
      IF STP(1)(X, q, T) GOTO E
      T ← T + 1
      GOTO A
```

```
[C]  Y ← 1
```

For a given input X , this program first simulates zero steps of the computation by the program with number p on input X and tests whether it terminates (only the empty program would terminate after zero steps). If so, the program above terminates with output one. If not, the program does the same simulation for program number q . If it terminates, the program above terminates with output zero, otherwise, the same as above is repeated, but this time one step is simulated. The number of simulated steps increases until either p or q terminates. At any point, if p terminates, the output is one, and if q terminates, the output is zero.

This program is useful if we know that either p or q will terminate on input X and want to determine which one it is. We cannot simulate the entire program execution of p and then the execution of q , because p may never terminate, and then our program would run forever without providing any conclusive result. We used this technique, called “dovetailing,” to show that whenever B and $\neg B$ are r.e., then B is recursive.

Question 5: Enumeration

Explain in detail in your own words why the following theorem is true:

“A set B is r.e. if and only if there is an n for which $B = W_n$.”

W_n is the domain of the partially computable function computed by the program with number n :

$$W_n = \{ x \in N \mid \Phi(x, n) \downarrow \}$$

So the infinite list $W_0, W_1, W_2 \dots$ includes the domains of all partially computable functions.

Definition of r.e.: A set B is r.e. just when it is the domain of a partially computable function $g(x)$:

$$B = \{ x \in N \mid g(x) \downarrow \}$$

So B is r.e. if and only if B is in the list $W_0, W_1, W_2 \dots$, that is, if there is an n for which $B = W_n$.