Honors Theory, Spring 2002, Yap
Homework 3 Solution

DISCLAIMER ABOUT SOLUTION: In the interest of posting this in a timely fashion, I have not been able to fully debug the solution. But I believe the solution is substantially correct. –Chee.

MOSTLY ABOUT RECURSIVELY ENUMERABLE SETS In the following, we will identify the set $\{0,1\}^*$ with $\mathbb{N}$ via the dyadic notation. So we will interchangeably talk of (natural) numbers and binary strings.

1. (20 Points) In the notes on the Bernstein-Schröder theorem we showed a fixed point A^* for any monotone map $\mu : 2^X \to 2^X$. This fixed point was based on the idea of sets $A \subseteq X$ that are **small** in the sense that $A \subseteq \mu(A)$. (The set A is "small" relative to its image $\mu(A)$.) Now define a set $B \subseteq X$ to be **big** if $\mu(B) \subseteq B$. Show a fixed point B^* for μ based on big sets.

   SOLUTION: Define B^* to be the intersection of all the big sets. To show that B^* is a fixed point, we first show one direction:
   $$\mu(B^*) \subseteq B^*$$
   This amounts to saying that B^* is big. If $b \in \mu(B^*)$, by definition of B^*, $B^* \subseteq B$ for any big B, and so $b \in \mu(B^*) \subseteq \mu(B)$, by monotonicity. As B satisfies $\mu(B) \subseteq B$, we have $b \in B$, for any B that is big. But B^* is the intersection of all the B's that are big, thus $b \in B^*$. In the other direction, we make a general observation: if B is big, then $\mu(B)$ is big. This is because the bigness of B implies $\mu(\mu(B)) \subseteq \mu(B)$, by monotonicity. Since B^* is big, $\mu(B^*)$ must be big. As B^* is the intersection of all big sets, this shows $B^* \subseteq \mu(B^*)$.

   Extra Credit: Construct an example in which the $B^* \neq A^*$.

   SOLUTION: An example function: $\mu : 2^X \to 2^X, \mu(S) = S$ for any $S \in X$. The function is monotone as $\mu(S_1) \subseteq \mu(S_2)$, for any $S_1 \subseteq S_2 \subseteq X$. As each set is both small and big, $A^* = X$ is the union of all small sets, and $B^* = \emptyset$ is the intersection of all big sets. Thus $A^* \neq B^*$.

2. (15 Points) Prove that a set $A \subseteq \mathbb{N}$ is r.e. if and only if there exists a recursive set $B \subseteq \{0,1,\#\}^*$ such that $A = \{w \in \{0,1\}^* : (\exists y \in \{0,1\}^*)[w\#y \in B]\}$.

   SOLUTION: Let's show the direction from right to left first. As B is recursive, there is a STM M_B that decides B. We need to prove that the language $A = \{w \in \{0,1\}^* : (\exists y \in \{0,1\}^*)[w\#y \in B]\}$ is r.e.. We construct a STM M_A that works in the following way:

   M_A = "On input w:
   1. Repeat the following for i = 1,2,3,...
   2. Run M_B on $w\#s_i$.
   3. If M_B accepts, accept."

   Here $s_1, s_2, s_3...$ is a list of all possible strings in $\{0,1\}^*$. We know that M_A recognizes A, because:

   For each $w \in A$, there exists a string y such that $w\#y \in B$. So after a finite number of tests, y will be found. As M_B is a decider, each test of y is done in finite steps. Thus w will be accepted by M_A in finite steps.
   For each $w \notin A$, there is no string y such that $w\#y \in B$. So M_A will try to find y for ever, so it loops.

   As STM M_A recognizes A, A is r.e..

   Then we prove the direction from left to right. Suppose A is r.e., then there is a STM M_A that recognizes A. we need to find a recursive language B such that $A = \{w \in \{0,1\}^* : (\exists y \in \{0,1\}^*)[w\#y \in B]\}$. Intuitively, we want B to accept $w\#C$ where C is the encoded string for configuration history of M_A running w. We construct a STM M_B that decides B:

   M_B = "On input w:
   1. If w is not of the form $x\#C_1 \to C_2 \to ...C_m, m \geq 1$ (where symbol $\to$ separates the configurations C_i's), rejects.
   2. Repeat stage 3 for i = 1 to m
   3. Run M_A on w for the ith step. If M_A halts and rejects, reject; Else if the current configuration of

M_A is not the same as C_i also reject. Else if $i = m$ and M_A accepts w, accept."
4. If M_A still does not accept after m steps (C_m is not an accepting configuration), reject.

STM M_B is a decider as it always halts in finite simulation steps of M_A. Obviously it only accepts those strings of form $w\#C$ where C is the encoded string of accepting configuration history of M_A running w. If $w \in A$, there exists an accepting configuration history y of M_A running w, so M_B accepts $w\#y$; If $w \notin A$, there is no accepting configuration history of M_A running w, M_B rejects $w\#y$ for all y's.

3. (15+15+10 Points) Fix a deterministic universal Turing machine U such that $K(U) = RE|\{0,1\}$. We can view this U as computing a function $\Phi : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$, where the output alphabet of U is assumed to be $\{0,1\}$ and we identify the set $\{0,1\}^*$ with $\mathbb{N}$ via the dyadic notation. Recall how transducers define functions – on input $\langle i, w\rangle$, if v is the non-blank word that is being scanned by the work head when U enters the accept state q_A, then $\Phi(i,w)$ is the dyadic number v. If $U(i,w)\uparrow$, then $\Phi(i,w)$ is undefined. Define

$$\Phi := \{\phi_i : i \in \mathbb{N}\}$$

where $\phi_i : \mathbb{N} \to \mathbb{N}$ be the function $\phi_i(w) = \Phi(i,w)$. We define a function f to be **partial recursive** if $f \in \Phi$. Thus Φ is the function analogue of the class RE. Let sets $W_i := \{w \in \mathbb{N} : \phi_i(w)\downarrow\}$ and $E_i := \{\phi_i(w) : w \in \mathbb{N}, \phi_i(w)\downarrow\}$. Thus, W_i, E_i are basically the domain and range of ϕ (Mnemonic: ϕ_i is a map from the "west" set W_i to the "east" set E_i.) Prove the following:
(i) A set $A \subseteq \mathbb{N}$ is r.e. iff $A = W_i$ for some i.
(ii) A set $B \subseteq \mathbb{N}$ is r.e. iff $B = E_i$ for some i. HINT: to show that E_i is r.e. you need to "dovetail" together a denumerable sequence of computations.
(iii) The set $TOT := \{i \in \mathbb{N} : \phi_i \text{ is total}\}$ is not r.e.

SOLUTION:

(i) Let's prove the direction from left to right first. As the set A is r.e., there is a STM M that recognizes it. We construct a function ϕ_a for it.

$\phi_a =$ "On input w
1. Run M on w.
2. If M accepts w, halt and output w. Else if M halts and rejects w, loop."

For each $w \in A$, ϕ_a halts; For each $w \notin A$, ϕ_a is undefined. So $A = \{w \in N : \phi_a(w)\downarrow\}$.

Now prove the direction from right to left. This is trivial. If $A = W_i$ for some i, we can construct a STM M from ϕ_i. Compute $\phi_i(w)$, whenever ϕ_i halts, M accepts w. If ϕ_i does not halt, M does not halt either. Since M accepts only all the strings in A, it recognizes A. Thus A is r.e..

(ii) Let's prove the direction from left to right first. As the set A is r.e., there is a STM M that recognizes it. We construct a function ϕ_a the same way as part (i).

$\phi_a =$ "On input w
1. Run M on w.
2. If M accepts w, halt and output w. Else if M halts and rejects w, loop."

For each $w \in A$, ϕ_a halts and outputs w; For each $w \notin A$, ϕ_a is undefined. So $A = \{\phi_a(w) = w : w \in N, \phi_a(w)\downarrow\}$.

Now prove the direction from right to left. If $A = E_i$ for some i, we can construct a STM M from ϕ_i:

M = "On input w
1. Repeat the following for i = 1, 2, 3 ...
2. Compute ϕ_i for i steps on each string $s_1, s_2, ...s_i$.
3. If any computation halts and outputs w, halt and accept."

Where $s_1, s_2, ...s_i$ is a list of all possible strings.

For each string $w \in E_i$, on some string u, ϕ_i halts and outputs w, so M will accept w in finite steps. For each string $w \notin E_i$, ϕ_i will never find any string u, such that $\phi_i(u) = w$, so M will loop for ever. Thus STM M recogizes A.

(iii)Suppose not, the set TOT is r.e.. There is a recursive enumerator E enumerates it. Consider the total function $g(m) = \phi_m(m) + 1$ for each $m \in N$. Obviously g is computable. But as g differs from each function ϕ_i, it is not included in the enumeration E. That's a contradiction. Thus the set TOT is not r.e..