Final Exam

Aaron Rosen

Monday, May 13

Problem 1

$\mathbb{F} = \{0^x : x \text{ is the } n^{th} \text{ fibonacci number with } n \geq 0\}$

Consider the following strings:

$0^1, 0^2, 0^3 \in \mathbb{F}$

Concatenating 0 we get:

$0^1 0 = 0^2 \in \mathbb{F}$
$0^2 0 = 0^3 \in \mathbb{F}$
$0^3 0 = 0^4 \notin \mathbb{F}$

Now for any k and any $j > 0$

$0^k 0^j \notin \mathbb{F}$ unless $j = k - 1$

Therefore

$0^{k+j} \not\equiv_R 0^k 0^j$ for all k and j

So all members of $\mathbb{F}$ except 0^1 and 0^2 are pairwise non-equivalent, hence

$|{}^{\Sigma}/_{\not\equiv_R}| = \infty$

So, by the Myhill-Nerode Theorem, $\mathbb{F}$ is not regular.

Problem 2

We want to show an r.e. set is recursive iff an enumeration machine enumerates it in increasing order.

$\Leftarrow$ Say A is enumerated by an enumeration machine E in increasing order. Let a TM M, accept A. On input x, A is enumerated by E and M waits to see if x comes up. If it does, M halts and enters an accept state. If a number greater then x comes up, M enters a reject state and halts. Since A is enumerated in increasing order, M will always either halt and accept or halt and reject so it is recursive.

$\Rightarrow$ Suppose A is recursive, that is, there is a total TM M such that $A = L(M)$. Since A is recursive, M either accepts or rejects, but it always halts. We define M as always accepting as long as it is given numbers in increasing order. Therefore the set enumerated in increasing order will be accepted.

Problem 3

We want to show A is r.e. iff there is a recursive binary relation R such that $A = \{x : \exists y(R(x, y))\}$

$\Rightarrow$ Assume A is r.e. This means there is some machine, we will call it M_A such that $A = L(M_A)$. First we will define a binary relation R as follows:

$$(x, y) \in R \iff M_A(x) \downarrow a \text{ in } y \text{ steps} \iff M_A^y(x) \downarrow a$$

Now with this R we get

$$A = \{x : \exists y(M_A^y(x) \downarrow a)\}$$

Which means A consists of all strings x accepted by M_A in a finite number of steps. Since A is r.e. this is exactly what we want. And there is recursive binary relation R such that $A = \{x : \exists y(R(x, y))\}$

$\Leftarrow$ Assume there is a recursive binary relation R such that $A = \{x : \exists y(R(x, y))\}$.

We know the set $\{x\#y : R(x,y)\}$ is recursive. Then there is a TM M_R which, given $x\#y$ either accepts or rejects

$$M_R(x\#y) = \begin{cases} M_R(x\#y) \downarrow a, & \text{if } (x,y) \in R \\ M_R(x\#y) \downarrow r, & \text{if } (x,y) \notin R \end{cases}$$

Now we will build a TM M_A that simulates M_R such that

$(x,y) \in R \Rightarrow M_R(x\#y) \downarrow a \Rightarrow M_A(x) \downarrow a$
$(x,y) \notin R \Rightarrow M_R(x\#y) \downarrow r \Rightarrow M_A(x)$ does not accept

So given any x we can try all y's until we find one such that

$M_R(x\#y)$ accepts

and if so $M_A(x)$ accepts. Otherwise, if we never find a y such that M_R accepts, then $M_A(x)$ diverges. Hence A is r.e.

Problem 4

$TOT = \{M : M \text{ is a total TM}\}$

First we will show that $\overline{TOT}$ is not r.e. using the reduction

$\overline{HP} \leqslant_m^\sigma \overline{TOT}$

We know $\overline{HP}$ is not r.e. because HP is r.e. and not recursive. Now we need a σ such that

$M\#x \in \overline{HP} \Rightarrow \sigma(M\#x) \in \overline{TOT}$
$M\#x \notin \overline{HP} \Rightarrow \sigma(M\#x) \notin \overline{TOT}$

Say we use $\sigma(M\#x)$ runs M on input x

$$
\begin{aligned}
M\#x \in \overline{HP} &\Rightarrow M(x)\uparrow \\
&\Rightarrow \sigma(M\#x) \text{ diverges on all inputs} \\
&\Rightarrow \sigma(M\#x) \in \overline{TOT} \\
M\#x \notin \overline{HP} &\Rightarrow M(x)\downarrow \\
&\Rightarrow \sigma(M\#x) \text{ halts on all inputs} \\
&\Rightarrow \sigma(M\#x) \notin \overline{TOT}
\end{aligned}
$$

Therefore $\overline{HP}$ reduces to $\overline{TOT}$ and $\overline{TOT}$ is not r.e.

Now we will show TOT is not r.e.
Assume, for the sake of contradiction, that TOT is r.e. Then there exists a TM K such that

$$L(K) = TOT$$

Thus on input $M\#x$

$$K(M\#x) = \begin{cases} \downarrow a, & \text{if } M(x) \text{ halts} \\ \downarrow r \text{ or} \uparrow, & \text{if } M(x) \text{ diverges} \end{cases}$$

Now consider a TM N defined as

$$N(x) = \begin{cases} \downarrow a, & \text{if } K(M_x\#x) \text{ does not accept} \\ \downarrow r, & \text{if } K(M_x\#x)\downarrow a \end{cases}$$

So for any x

$$
\begin{aligned}
N(x)\downarrow a &\text{ iff } K(M_x\#x) \text{ does not accept} \\
&\text{ iff } M_x(x) \text{ diverges}
\end{aligned}
$$

Therefore $N(x)$ was not enumerated by E, and we've reached a contradiction.
So TOT is not r.e.

Problem 5

$L = \{M : M \text{ accepts at least } n \text{ strings }\}$

First we will show L is r.e. by building a TM M_L that accepts it. M_L takes as input a machine, we'll call it Q, and a list of binary strings, delimited by the dollar sign \$ in increasing order of size (i.e. \$0\$1\$00\$01\$11\$000\$...) for each string x in the list of binary strings Q will run for $|x|$ steps on x and every string before it. It will continue to do this until Q accepts at least n strings at which point M_L will accept it. If Q never accepts n strings then M_L will loop. Therefore, we have shown that L is r.e.

Now we will show that $\overline{L}$ is not r.e. by reducing a non-r.e. set to it. We can use the reduction

$$\overline{MP} \leqslant_m^\sigma \overline{L}$$

Let $\sigma(M\#x)(y)$ ignore the input y and run M on x. If $M(x)$ accepts then $\sigma(M\#x)(y)$ accepts, and if $M(x)$ does not accept then $\sigma(M\#x)(y)$ does not accept.

$$\begin{aligned}
M\#x \in \overline{MP} &\Rightarrow M(x) \text{ does not accept} \\
&\Rightarrow L(\sigma(M\#x)) = \emptyset \\
&\Rightarrow L(\sigma(M\#x)) \text{ has fewer than } n \text{ elements} \\
&\Rightarrow \sigma(M\#x) \in \overline{L} \\
M\#x \notin \overline{MP} &\Rightarrow M(x) \downarrow a \\
&\Rightarrow L(\sigma(M\#x)) = \Sigma^* \\
&\Rightarrow L(\sigma(M\#x)) \text{ has more than } n \text{ elements} \\
&\Rightarrow \sigma(M\#x) \notin \overline{L}
\end{aligned}$$

And since $\overline{MP}$ is not r.e. $\overline{L}$ must not be r.e.

*Problem 6

Let A and B be r.e. and $A/B = \{x : \exists y \in B(xy \in A)\}$. There are machines M_A and M_B such that $A = L(M_A)$ and $B = L(M_B)$

Now we will design a machine M that will accept the set A/B. M simulates the instructions of M_A and M_B. Given an x, M will dovetail the operation of finding a y in B such that $xy \in A$. Since B is r.e. there is an enumeration machine E_B that enumerates all of the elements of B. Going through all of the elements of this enumeration, for the n^{th} element in $Enum(E_B)$, run M_A for n steps on xy_i $(0 < i \leq n)$.

$M(x)$ accepts if $M_A^n(xy_i)$ accepts for one of these y_is
$M(x)$ rejects otherwise

Therefore A/B is r.e.

*Problem 7

(1)

Assume, for the sake of contradiction $L(M) = L(N)$ is decidable. Then there is a total machine Q such that

$$Q(M\#N) = \begin{cases} \downarrow a, & \text{if } L(M) = L(N) \\ \downarrow r, & \text{if } L(M) \neq L(N) \end{cases}$$

Let $L(N) = \{x\}$, then

$$Q(M\#N) \downarrow a \Rightarrow L(M) = \{x\} \Rightarrow x \in L(M) \Rightarrow M\#x \in MP$$

But MP is undecidable, so $L(M) = L(N)$ is undecidable.

(2)

Assume, for the sake of contradiction $L(M) \subseteq L(N)$ is decidable. Then there is a total machine Q such that

$$Q(M\#N) = \begin{cases} \downarrow a, & \text{if } x \in L(M) \Rightarrow x \in L(N) \\ \downarrow r, & \text{otherwise} \end{cases}$$

Let $L(M) = x$, then

$$Q(M\#N) \downarrow a \Rightarrow x \in L(N) \Rightarrow N\#x \in MP$$

But MP is undecidable, so $L(M) \subseteq L(N)$ is undecidable.

(3)

Assume, for the sake of contradiction $L(M) \cap L(N) = \emptyset$ is decidable. Then there is a total machine Q such that

$$Q(M\#N) = \begin{cases} \downarrow a, & \text{if } x \in L(M) \Rightarrow x \notin L(N) \\ \downarrow r, & \text{otherwise} \end{cases}$$

Let $L(N) = x$, then

$$Q(M\#N) \downarrow a \Rightarrow x \notin L(M) \Rightarrow M\#x \notin MP$$

But MP is undecidable, so $L(M) \cap L(N) = \emptyset$ is undeciable.