Lecture Notes, Math 497A, Fall 2007: Computability, Unsolvability, Randomness

Stephen G. Simpson
Department of Mathematics
Pennsylvania State University
http://www.math.psu.edu/simpson/

December 13, 2007

Contents

Lecture 1: August 27, 2007

1 Computable functions

We denote the natural numbers by $\mathbb{N} = \{0, 1, 2, \ldots\}$. We shall deal with number-theoretic functions $f : \mathbb{N} \to \mathbb{N}$.

Example 1.1. An example of a 1-place number-theoretic function is $f(x) = 2^x$. Thus $f(0) = 1$, $f(1) = 2$, $f(2) = 4$, $f(3) = 8$, etc. Note that the variable x ranges over $\mathbb{N}$, i.e., x takes on integer values only.

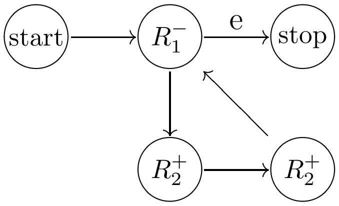
In order to rigorously define what we mean by a computable function, we introduce *register machines.* A register machine consists of a finite set of registers, $R_1, \ldots, R_s$. Each register R_i can hold an arbitrary natural number $z_i \in \mathbb{N}$. Here $1 \leq i \leq s$. We think of register R_i as a box containing z_i marbles. The basic register machine operations are adding a marble to a box and removing a marble from a register. The register R_i is said to be *empty* if $z_i = 0$. A *register machine program* is a connected flow diagram consisting of the following types of instructions.

1. The *increment* instruction, denoted R_i^+. This instruction replaces z_i by $z_i + 1$ (i.e., adds a marble to the box R_i) and then goes to the next instruction, indicated by an arrow.

2. The *decrement* instruction, denoted R_i^-. This is a branching instruction. If $z_i > 0$, it replaces z_i by $z_i - 1$ (i.e., removes a marble from the box R_i) and then goes to the next instruction, indicated by an unlabeled arrow. If $z_i = 0$, it leaves R_i empty and goes to the next instruction, indicated by an arrow labeled e.

3. Start and stop instructions. A program has exactly one start instruction. A stop instruction indicates that we are to stop, i.e., no more instructions are to be executed.

Example 1.2. Consider the following register machine program.



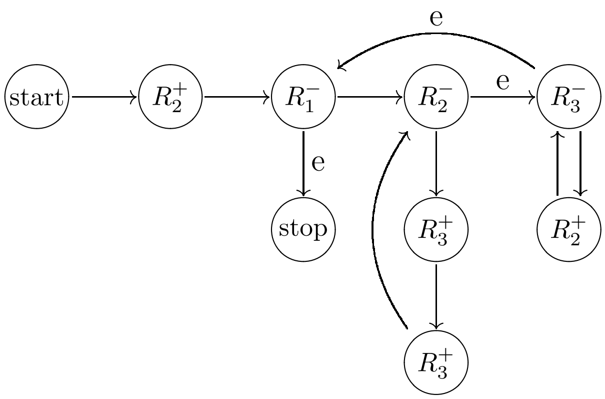
Note that this program, if started with x in R_1 and y in R_2, will eventually halt (after a finite number of steps) with 0 in R_1 and $2x + y$ in R_2.

Definition 1.3 (computable functions). A 1-place number-theoretic function $f(x)$ is said to be *computable* if there exists a program, call it $\mathcal{P}$, with the following property. For any $x \in \mathbb{N}$, if we start $\mathcal{P}$ with x in R_1 and all other registers empty, i.e., 0 in $R_2, \ldots, R_s$, then $\mathcal{P}$ will eventually halt with $f(x)$ in R_2.

Example 1.4. The program of Example 1.2 computes the function $f(x) = 2x$. Thus, $f(x) = 2x$ is a computable function.

Exercise 1.5. Exhibit a register machine program showing that the function $f(x) = 2^x$ is computable.

Solution. The following program, if started with x in R_1 and 0 in R_2 and R_3, will eventually halt with 2^x in R_2. This shows that the function 2^x is computable.



□

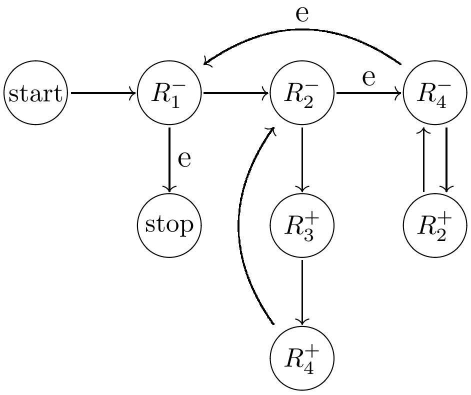
We generalize Definition 1.3 as follows.

Definition 1.6 (computable functions). A k-place number-theoretic function, $f : \mathbb{N}^k \to \mathbb{N}$, is said to be *computable* if there exists a program, call it $\mathcal{P}$, with the following property. For any $x_1, \ldots, x_k \in \mathbb{N}^k$, if we start $\mathcal{P}$ with x_1 in R_1, …, x_k in R_k, and all other registers empty, i.e., 0 in $R_{k+1}, \ldots, R_s$, then $\mathcal{P}$ will eventually halt with $f(x_1, \ldots, x_k)$ in R_{k+1}.

Notation 1.7. We write $\mathcal{P}(x_1, \ldots, x_k)$ to denote the run of the program $\mathcal{P}$ when it is started with x_1 in R_1, …, x_k in R_k, and all other registers empty. In this case $R_1, \ldots, R_k$ are *input registers*, R_{k+1} is the *output register*, and $R_{k+2}, \ldots, R_s$ are *auxiliary registers.*

Exercise 1.8. Exhibit a register machine program showing that the 2-place number-theoretic function $f(x, y) = xy$ is computable.

Solution. The following program, if started with x in R_1, y in R_2, and 0 in R_3, will eventually halt with xy in R_3.



Note that R_4 serves as an auxiliary register. □

Exercise 1.9. Exhibit a register machine program which computes the exponential function, $\exp(x, y) = x^y$. Note that $x^0 = 1$ for all x, even for $x = 0$.

Exercise 1.10. Exhibit a register machine program which computes the function $\mathrm{Rem}(x, y) =$ the remainder of x on division by y.

Exercise 1.11. Assume that $P(x, y, z)$ is a 3-place predicate which is computable. Consider the 2-place partial function $\psi(x, y)$ defined as follows:

> $\psi(x, y) \simeq$ the least z such that $P(x, y, z)$ holds, if such a z exists. If such a z does not exist, then $\psi(x, y)$ is undefined.

Use register machine programs to prove that the function ψ is computable. (More precisely, ψ is partial recursive.)

Hint: Since P is a computable predicate, we may assume that we have a program $\mathcal{P}$ which computes the characteristic function of P. Show how to embed $\mathcal{P}$ (or perhaps a variant of $\mathcal{P}$) into a larger program, call it $\mathcal{Q}$, such that $\mathcal{Q}$ computes ψ. The idea of $\mathcal{Q}$ is that, given x and y, $\mathcal{Q}(x, y)$ searches sequentially through the integers $z = 0$, $z = 1$, $z = 2$, …, to find the first z such that $P(x, y, z)$ holds. In particular, $\mathcal{Q}$ will have the property that, for all x and y, $\mathcal{Q}(x, y)$ eventually halts if and only if $P(x, y, z)$ holds for some z.

Exercise 1.12. Use the results of Exercises 1.10 and 1.11 to prove that the 2-place function $\mathrm{LCM}(x, y) =$ the least common multiple of x and y is computable. Deduce that the 2-place function $\mathrm{GCD}(x, y) =$ the greatest common divisor of x and y, is also computable.

Later we shall prove the following important theorem.

Theorem 1.13 (Turing, 1936). We can construct a particular program, call it $\mathcal{P}$, with the following property. The 1-place function

$$g(x) = \begin{cases} 1 & \text{if } \mathcal{P}(x) \text{ eventually halts,} \\ 0 & \text{if } \mathcal{P}(x) \text{ never halts,} \end{cases}$$

is not computable. (Here we are using Notation 1.7.)

In other words, the Halting Problem for $\mathcal{P}$ is unsolvable.

2 Homework #1, due September 4, 2007

Exercises 2.1.

1. Exhibit a register machine program which computes the exponential function, $\exp(x, y) = x^y$. Remember that the variables x and y range over $\mathbb{N}$, the set of natural numbers. Note that $x^0 = 1$ for all x, even for $x = 0$.

2. Exhibit a register machine program which computes the function

$$\text{Rem}(y, x) = \text{the remainder of } y \text{ on division by } x.$$

For example, Rem(17,5)=2.

3. Assume that $P(x, y, z)$ is a 3-place predicate which is computable. Consider the 2-place partial function $\psi(x, y)$ defined as follows:

$\psi(x, y) \simeq$ the least z such that $P(x, y, z)$ holds, if such a z exists.
If such a z does not exist, then $\psi(x, y)$ is undefined.

Use register machine programs to prove that the function ψ is computable. (More precisely, ψ is partial recursive.)

Hint: Since P is a computable predicate, we may assume that we have a program $\mathcal{P}$ which computes the characteristic function of P. Show how to embed $\mathcal{P}$ (or perhaps a variant of $\mathcal{P}$) into a larger program, call it $\mathcal{Q}$, such that $\mathcal{Q}$ computes ψ. The idea of $\mathcal{Q}$ is that, given x and y, $\mathcal{Q}(x, y)$ searches sequentially through the integers $z = 0$, $z = 1$, $z = 2$, …, to find the first z such that $P(x, y, z)$ holds. In particular, $\mathcal{Q}$ will have the property that, for all x and y, $\mathcal{Q}(x, y)$ eventually halts if and only if $P(x, y, z)$ holds for some z.

4. Use the results of Problems 2 and 3 to prove that the 2-place function $\text{LCM}(x, y) =$ the least common multiple of x and y is computable. Deduce that the 2-place function $\text{GCD}(x, y) =$ the greatest common divisor of x and y, is also computable.

5. (Approximating the square root of 2.) Consider successive rational approximations of $\sqrt{2}$ given by Newton's method:

$$x_0 = 1, \qquad x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

where $f(x) = x^2 - 2$. The first few values are $x_0 = 1$, $x_1 = 3/2$, $x_2 = 17/12$, $x_3 = 577/408$. Let $a(n)$ and $b(n)$ respectively be the numerator and denominator of x_n. Thus $a(n)$ and $b(n)$ are 1-place number-theoretic functions. The first few values are $a(0) = b(0) = 1$, $a(1) = 3$, $b(1) = 2$, $a(3) = 17$, $b(3) = 12$, $a(3) = 577$, $b(3) = 408$. Use primitive recursion to prove that the functions $a(n)$ and $b(n)$ are computable.

6. A positive real number r is said to be *computable* if there exist computable sequences of positive integers a_n, b_n, $n = 0, 1, 2, \ldots$, such that

$$r = \lim_{n\to\infty} \frac{a_n}{b_n}$$

and in addition

$$\left| r - \frac{a_n}{b_n} \right| < \frac{1}{2^n}$$

for all $n \in \mathbb{N}$. Give a convincing argument that all of the standard examples of positive real numbers including $\sqrt{2} = 1.41421\cdots$, $e = 2.71828\cdots$, $\pi = 3.14159\cdots$, etc., are computable.

7. Prove that the sum, product, and quotient of two computable positive real numbers are computable.

Lecture 2: August 29, 2007

3 Primitive recursion

An important tool for proving that many familiar functions are computable is primitive recursion. This is essentially just the well-known method of "definition by induction."

Example 3.1. We can define the factorial function $f(n) = n!$ by

$$\begin{aligned} 0! &= 1 \\ (n+1)! &= n! \cdot (n+1). \end{aligned}$$

Note that there is one and only one (unique) number-theoretic function f satisfying these equations.

Example 3.2. Similarly, we can define multiplication by

$$\begin{aligned} f(x,0) &= 0 \\ f(x,y+1) &= f(x,y)+x \end{aligned}$$

because the only number-theoretic function satisfying these equations is $f(x,y) = xy$. In these terms the equations look like this:

$$\begin{aligned} x \cdot 0 &= 0 \\ x \cdot (y+1) &= x \cdot y + x. \end{aligned}$$

Lemma 3.3 (primitive recursion). Let $g(x_1,\ldots,x_k)$ be a k-place number-theoretic function, and let $h(y,z,x_1,\ldots,x_k)$ be a $(k+2)$-place number-theoretic function. Then, there is one and only one $(k+1)$-place number-theoretic function $f(y,x_1,\ldots,x_k)$ satisfying the *primitive recursion equations*

$$\begin{aligned} f(0,x_1,\ldots,x_k) &= g(x_1,\ldots,x_k), \\ f(y+1,x_1,\ldots,x_k) &= h(y,f(y,x_1,\ldots,x_k),x_1,\ldots,x_k) \end{aligned}$$

for all $y, x_1, \ldots, x_k$. Moreover, if g and h are computable, then f is computable.

Proof sketch. The program for f includes the programs for g and h and works as follows. Given the inputs $y, x_1, \ldots, x_k$, we wish to compute $f(y,x_{,},\ldots,x_k)$. To guide the computation, we maintain a counter i which will run from 0 to

y. We also maintain a quantity called z. At certain stages of the computation, z will take on successive values $z_i = f(i, x_1, \ldots, x_k)$ where $i = 0, 1, \ldots, y$. We compute these values as follows. Initially $i = 0$ and we use our program for g to compute $z_0 = g(x_1, \ldots, x_k)$. Then, for each $i < y$ successively, having already computed z_i we increment the counter i and use our program for h to compute $z_{i+1} = h(i, z_i, x_1, \ldots, x_k)$. Finally, when $i = y$, we output $z = z_y = f(y, x_1, \ldots, x_k)$.

For more details, see Lemma 1.3.4 and Figure 1.6 of the 558 notes. □

Example 3.4. An example illustrating the above proof sketch is the program for multiplication using repeated addition. See Exercise 1.8 and Example 3.2.

Exercise 3.5 (approximating the square root of 2). Consider successive rational approximations of $\sqrt{2}$ given by Newton's method

$$x_0 = 1, \qquad x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

where $f(x) = x^2 - 2$. The first few values are $x_0 = 1$, $x_1 = 3/2$, $x_2 = 17/12$, $x_3 = 577/408$. Let $a(n)$ and $b(n)$ respectively be the numerator and denominator of x_n. Thus $a(n)$ and $b(n)$ are 1-place number-theoretic functions. The first few values are $a(0) = b(0) = 1$, $a(1) = 3$, $b(1) = 2$, $a(3) = 17$, $b(3) = 12$, $a(3) = 577$, $b(3) = 408$. Use primitive recursion to prove that the functions $a(n)$ and $b(n)$ are computable.

Exercise 3.6. A positive real number r is said to be *computable* if there exist computable sequences of positive integers a_n, b_n, $n = 0, 1, 2, \ldots$, such that

$$r = \lim_{n \to \infty} \frac{a_n}{b_n}$$

and in addition

$$\left| r - \frac{a_n}{b_n} \right| < \frac{1}{2^n}$$

for all $n \in \mathbb{N}$.

1. Use primitive recursion to give a convincing argument that all of the standard examples of positive real numbers including $\sqrt{2} = 1.41421\cdots$, $e = 2.71828\cdots$, $\pi = 3.14159\cdots$, etc., are computable.

2. Prove that the sum, product, and quotient of two computable positive real numbers are computable.

The following technical lemma will be useful.

Lemma 3.7 (iterated sum and product). Let $f(x, y, -)$ be a $(k+2)$-place function. Consider the $(k+1)$-place functions

$$g(y, -) = \sum_{x=0}^{y-1} f(x, y, -)$$

and

$$h(y,-) = \prod_{x=0}^{y-1} f(x,y,-)$$

If f is computable, then g and h are computable.

Proof. We use primitive recursion to obtain $(k+2)$-place functions

$$g^*(w,y,-) = \sum_{x=0}^{w-1} f(x,y,-)$$

and

$$h^*(w,y,-) = \prod_{x=0}^{w-1} f(x,y,-)\,,$$

namely

$$\begin{array}{lcl} g^*(0,y,-) & = & 0 \\ g^*(w+1,y,-) & = & g^*(w,y,-) + f(w,y,-) \end{array}$$

and

$$\begin{array}{lcl} h^*(0,y,-) & = & 1 \\ h^*(w+1,y,-) & = & h^*(w,y,-) \cdot f(w,y,-)\,. \end{array}$$

By Lemma 3.3 g^* and h^* are computable. It follows that $g(y,-) = g^*(y,y,-)$ and $h(y,-) = h^*(y,y,-)$ are computable. □

4 Computable predicates

Definition 4.1. A k-place *predicate* is a set $P \subseteq \mathbb{N}^k$. We view P as a proposition with k variables:

$$P(x_1,\ldots,x_k) \equiv \text{“the k-tuple $\langle x_1,\ldots,x_k\rangle$ is an element of the set P.”}$$

If P is a k-place predicate, the *characteristic function* of P is the k-place function χ_P defined by

$$\chi_P(x_1,\ldots,x_k) = \begin{cases} 1 & \text{if } P(x_1,\ldots,x_k) \text{ is true,} \\ 0 & \text{if } P(x_1,\ldots,x_k) \text{ is false.} \end{cases}$$

We often identify P with χ_P. In particular, a predicate is said to be *computable* if and only if its characteristic function is computable.

Example 4.2. Consider the 2-place predicate $P(x,y) \equiv$ “x is a divisor of y.” Note that $P(3,5)$ is false, while $P(3,6)$ is true. Viewing this predicate as a set $P \subseteq \mathbb{N}^2$, we have $P = \{\langle x,y\rangle \mid x \text{ is a divisor of } y\}$. Thus $\chi_P(x,y) = 1$ if x is a divisor of y, $\chi_P(x,y) = 0$ otherwise. It can be shown that the 2-place function χ_P is computable. Therefore, the predicate P is computable.

Lemma 4.3 (Boolean operations). Assume that P and Q are computable k-place predicates. Then, the following Boolean combinations are computable:

1. $P \wedge Q \equiv$ "P and Q", in other words, $P \cap Q$.
2. $P \vee Q \equiv$ "P or Q", in other words, $P \cup Q$.
3. $\neg P \equiv$ "not P", in other words, $\mathbb{N}^k \setminus P$, the complement of P.

Proof. First, $P \wedge Q$ is computable because $\chi_{P \wedge Q}(-) = \chi_P(-) \cdot \chi_Q(-)$. Next, $\neg P$ is computable because $\chi_{\neg P}(-) = \alpha(\chi_P(-))$ where $\alpha(1) = 0$, $\alpha(0) = 1$. Finally, $P \vee Q$ is computable because $\chi_{P \vee Q}(-) = \alpha(\alpha(\chi_P(-)) \cdot \alpha(\chi_Q(-)))$. □

Lemma 4.4 (bounded quantification). Assume that $R(x, y, -)$ is a $(k+2)$-place predicate. Define $(k+1)$-place predicates

$$P(y, -) \ \equiv \ (\forall x < y)\, R(x, y, -)$$

and

$$Q(y, -) \ \equiv \ (\exists x < y)\, R(x, y, -)\,.$$

If R is computable, then P and Q are computable.

Proof. We have

$$\chi_P(y, -) = \prod_{x=0}^{y-1} \chi_R(x, y, -)$$

and

$$\chi_Q(y, -) = \alpha\left(\prod_{x=0}^{y-1} \alpha(\chi_R(x, y, -)\right)$$

so our result follows by Lemma 3.7. □

Remark 4.5. The content of the two previous lemmas is that the class of computable predicates is closed under Boolean operations and bounded quantification. This provides an easy way to prove that many familiar predicates are computable.

Example 4.6. Consider the 1-place predicate

$$\text{Prime}(x) \ \equiv \ \text{“}x \text{ is a prime number.”}$$

This predicate can be written using Boolean operations and bounded quantifiers as follows:

$$\text{Prime}(x) \ \equiv \ x > 1 \wedge \neg\, (\exists u < x)\, (\exists v < x)\, [\, x = u \cdot v\,]\,.$$

Therefore, by Lemmas 4.3 and 4.4, Prime(x) is a computable predicate. Note that we proved this without actually having to write a program to compute the characteristic function of the predicate Prime(x). Of course, we know in principle how to write such a program if needed.

Lecture 3: August 31, 2007

We now introduce the bounded least number operator, also known as the bounded μ-operator.

Lemma 4.7 (bounded μ-operator). Let $R(x, y, -)$ be a $(k+2)$-place computable predicate. Then the $(k+1)$-place function

$$\begin{aligned} f(y,-) &= (\mu x < y)\, R(x,y,-) \\ &= \begin{cases} \text{least } x \text{ such that } x < y \text{ and } R(x,y,-) \text{ holds,} \\ \qquad\qquad \text{if such an } x \text{ exists,} \\ y \text{ otherwise} \end{cases} \end{aligned}$$

is computable.

Proof. The idea of computing $f(y, -) = (\mu x < y)\, R(x, y, -)$ is as follows. First look at $x = 0$ and use the program for R to test whether $R(0, y, -)$ holds. If yes, output 0 and we are done. Otherwise, look at $x = 1$ and use the program for R to test whether $R(0, y, -)$ holds. If yes, output 1 and we are done. Otherwise, look at $x = 2, \ldots$. Continue in this fashion for y steps corresponding to $x = 0, 1, 2, \ldots, y-1$. If we reach the end of this process without finding an $x < y$ such that $R(x, y, -)$ holds, then output y.

Formally, we can prove the lemma by means of iterated sums and products, similar to the proof of Lemma 4.4. We have

$$\begin{aligned} f(y,-) &= \sum_{x=0}^{y-1} x \cdot \chi_R(x,y,-) \cdot \prod_{w=0}^{x-1} \alpha(\chi_R(w,y,-)) \\ &\quad + \; y \cdot \prod_{x=0}^{y-1} \alpha(\chi_R(x,y,-))\,. \end{aligned}$$

The first part of the above expression gives the least $x < y$ such that $R(x, y, -)$ holds, if it exists. The second part gives y if no such x exists. □

Example 4.8. Recall from Example 4.6 that the predicate Prime(x) ≡ "x is a prime number" is computable. We can use the bounded μ-operator plus primitive recursion to show that the function p_n = "the nth prime number" is computable as a function of n. Note that $p_0 = 2$, $p_1 = 3$, $p_2 = 5$, $p_3 = 7$, $p_4 = 11, \ldots$. The basic idea is to use primitive recursion $p_0 = 2$, $p_{n+1} =$ the least prime $> p_n$. To make this work using the bounded μ-operator, we need a bound on p_{n+1} in terms of p_n. Some well known bounds of this kind are

1. Euclid: $p_{n+1} < p_n! + 2$.
2. Bertrand: $p_{n+1} < 2p_n$.
3. Hadamard: the prime number theorem.

Let us use Euclid's bound. By Lemma 4.7 the 2-place function $g(x,y) = (\mu w < y)\,(w > x \wedge w \text{ is prime})$ is computable. Hence the function $h(x) = g(x, x! + 2)$ is also computable, but by Euclid's bound this is just the first prime $> x$. Now we have the recursion $p_0 = 2$, $p_{n+1} = h(p_n)$ and this shows that p_n is computable as a function of n.

5 Prime power coding

Given a k-tuple $\langle a_1, \ldots, a_k \rangle \in \mathbb{N}^k$, we can "encode" the k-tuple as a single integer, $z \in \mathbb{N}$. Namely,

$$z \;=\; \prod_{i=1}^{k} p_i^{a_i}\,.$$

For example, the "code" of the 3-tuple $\langle 8, 9, 10 \rangle$ is the number $z = 3^8 5^9 7^{10}$.

Moreover, the Fundamental Theorem of Arithmetic tells us that every positive integer can be factored uniquely into prime powers. Thus, z can be "decoded" to recover the k-tuple $\langle a_1, \ldots, a_k \rangle$.

For us, the point is that these coding and decoding methods are computable! For decoding, consider the 2-place function $g(z,i) =$ the exponent of p_i in the prime power decomposition of z. This is computable, because

$$g(z,i) = (\mu x < z)\,(\mathrm{Rem}(z, p_i^{x+1}) \neq 0).$$

Here $\mathrm{Rem}(u,v) =$ the remainder of u on division by v, which is computable by Exercise 1.10.

Notation 5.1. We write

$$(z)_i = \text{the exponent of } p_i \text{ in the prime power decompsition of } z.$$

We have seen that $(z)_i$ is computable as a function of two variables, z and i.

The following example illustrates how to use prime power coding to prove that various functions are computable.

Example 5.2 (the Fibonacci sequence). Consider the sequence

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, \ldots$$

defined by

$$\begin{aligned} F_0 &= 0 \\ F_1 &= 1 \\ F_n &= F_{n-1} + F_{n-2} \quad \text{for } n \geq 2\,. \end{aligned}$$

Here F_n is the nth Fibonacci number. We would like to prove that F_n is computable as a function of n. Unfortunately, the above definition of F_n does not quite fit into our scheme of primitive recursion,

$$\begin{aligned} f(0, -) &= g(-) \\ f(y+1, -) &= h(y, f(y,-), -) \end{aligned}$$

(see Lemma 3.3). The difficulty is that F_n is defined by a 2-step recursion, while primitive recursion is a 1-step recursion.

In order to get around this difficulty, let $G_n = 3^{F_n}5^{F_{n+1}}$ = the "code" of the ordered pair $\langle F_n, F_{n+1}\rangle$. Thus $(G_n)_1 = F_n$ and $(G_n)_2 = F_{n+1}$. We then have

$$\begin{aligned} G_0 &= 5 \\ G_{n+1} &= 3^{(G_n)_2}5^{(G_n)_1+(G_n)_2} \end{aligned}$$

and this is a 1-step recursion. Thus, by Lemma 3.3, G_n computable as a function of n. It follows that $F_n = (G_n)_1$ is computable as a function of n.

Remark 5.3. Prime power coding can be used in the solution of Exercise 3.5. In that exercise, we had two number theoretic functions $a(n)$ and $b(n)$ defined by a simultaneous recursion of the form

$$\begin{aligned} a(0) &= 1 \\ b(0) &= 1 \\ a(n+1) &= f(a(n), b(n)) \\ b(n+1) &= g(a(n), b(n)) \end{aligned}$$

where f and g are computable 2-place functions. Again, this is not strictly speaking a primitive recursion. However, as in the previous example, we can convert it to a primitive recursion by considering the function

$$c(n) \;=\; 3^{a(n)}5^{b(n)}\,.$$

As another application of prime power coding, we now introduce course-of-values recursion.

Course-of-values recursion is a stronger variant of primitive recursion, in which $f(y,-)$ is defined by recursion on y in terms of the entire sequence of previous values, $f(0,-), f(1,-), \ldots, f(y-1,-)$. We can use prime power coding to convert this to an ordinary 1-step recursion. Namely, consider the *course-of-values function* $\widetilde{f}$ defined by

$$\widetilde{f}(y,-) \;=\; \prod_{x=0}^{y-1} p_x^{f(x,-)}\,.$$

We then have the following lemma.

Lemma 5.4 (course-of-values recursion). Assume that $h(y,z,-)$ is a $(k+2)$-place number-theoretic function. Then, there is a unique $(k+1)$-place number-theoretic function $f(y,-)$ defined by

$$f(y,-) \quad = \quad h(y, \widetilde{f}(y,-), -)\,.$$

If h is computable, then f is computable.

Proof. The course-of-values function $\widetilde{f}$ can be defined from h by primitive recursion, namely

$$\begin{aligned}\widetilde{f}(0,-) &= 1 \\ \widetilde{f}(y+1,-) &= \widetilde{f}(y,-)\cdot p_y^{h(y,\widetilde{f}(y,-),-)}\,.\end{aligned}$$

Since h is computable, it follows by Lemma 3.3 that $\widetilde{f}$ is computable. Hence

$$f(y,-) \quad = \quad (\widetilde{f}(y+1,-))_y$$

is computable. □

Example 5.5. The function

$$f(y,z) \quad = \quad z + \sum_{x=0}^{y-1} f(x,z)^z$$

is computable, because it is defined by course-of-values recursion.

Remark 5.6. Summarizing our discussion so far, we have introduced several tools for showing that various familiar number-theoretic functions and number-theoretic predicates are computable. Among our tools are:

1. primitive recursion
2. iterated sums and products
3. Boolean operations $\land$, $\lor$, $\neg$
4. bounded quantifiers $(\forall x < y)$, $(\exists x < y)$
5. the bounded μ-operator
6. prime power coding
7. course-of-values recursion

6 Homework #2, due September 10, 2007

Exercises 6.1.

1. Let r be a positive real number. Prove that r is computable if and only if the number-theoretic function

   $$f(n) = \text{the } n\text{th decimal digit of } r$$

   is computable.

2. Consider the 2-place computable number-theoretic function $f(x, y) = x + y$. Exhibit three different indices of f.

   (By an *index* of a partial recursive function, we mean the Gödel number of some program which computes the function.)

3. If f is a computable permutation of $\mathbb{N}$, prove that the inverse permutation f^{-1} is also computable.

   (Here $f^{-1}(y) = x$ if and only if $f(x) = y$. By a *computable permutation of* $\mathbb{N}$ we mean a computable 1-place function $f : \mathbb{N} \to \mathbb{N}$ which maps $\mathbb{N}$ one-to-one onto $\mathbb{N}$.)

4. Generalize the previous exercise as follows. Prove that if ψ is a 1-place partial recursive function which is one-to-one, then the inverse function ψ^{-1} is again partial recursive.

5. Consider the sets
$$K_n = \{x \in \mathbb{N} \mid \varphi_x^{(1)}(x) \simeq n\}$$
   where $n = 0, 1, 2, \ldots$. Show that the sets K_0 and K_1 are recursively inseparable. More generally, show that K_m and K_n are recursively inseparable for all m, n such that $m \neq n$.

   (Two sets $A, B \subseteq \mathbb{N}$ are said to be *recursively separable* if there exists a recursive function $f : \mathbb{N} \to \{0, 1\}$ such that $f(n) = 1$ for all $n \in A$, and $f(n) = 0$ for all $n \in B$. Otherwise, A and B are said to be *recursively inseparable*.)

6. Let $\psi(x)$ and $\theta(x)$ be 1-place partial recursive functions. We say that *ψ is reducible to θ* if there exists a 1-place total recursive function $h(x)$ such that $\psi(x) \simeq \theta(h(x))$ for all $x \in \mathbb{N}$. We refer to $h(x)$ as a *reduction function*, and we say that *h reduces ψ to θ*. We say that θ is *universal* if all 1-place partial recursive functions are reducible to θ.

   (a) Prove that the 1-place partial recursive function $\varphi_x^{(1)}(x)$ is universal.

   (b) Give some additional examples of 1-place partial recursive functions which are universal.

   (c) Prove that if θ is universal then the domain of θ is not recursive.
   (The *domain of* θ is defined to be the set $\mathrm{dom}(\theta) = \{x \mid \theta(x)\downarrow\}$.)

   (d) Construct a 1-place partial recursive function θ which is universal via linear reduction functions.

   (This means that each 1-place partial recursive function is reducible to θ by means of a reduction function which is linear. We say that $h(x)$ is *linear* if there exist constants a and b such that $h(x) = ax + b$ for all x.)

7. (Extra Credit). Prove that any two universal partial recursive functions θ_1 and θ_2 are *recursively isomorphic*. This means that there exists a computable permutation of $\mathbb{N}$, call it f, such that

$$\theta_1(x) \simeq y \quad \text{if and only if} \quad \theta_2(f(x)) \simeq f(y)$$

for all x and y.

Lecture 4: September 5, 2007

7 Partial recursive functions

Definition 7.1 (partial functions). A (k-place number-theoretic) *partial function*

$$\psi :\subseteq \mathbb{N}^k \to \mathbb{N}$$

is a function $\psi(x_1, \ldots, x_k)$ where $x_1, \ldots, x_k \in \mathbb{N}$. The value $y = \psi(x_1, \ldots, x_k)$ also belongs to $\mathbb{N}$, or it may be undefined. Thus, the domain of ψ is a subset of $\mathbb{N}^k$. If the domain of ψ is all of $\mathbb{N}^k$, we say that ψ is *total.*

Example 7.2. The partial function $\psi(x, y) = x/y$ is undefined for $y = 0$ or if $\mathrm{Rem}(x, y) \neq 0$. Thus, the domain of ψ is $\{\langle x, y\rangle \mid y \neq 0 \text{ and } \mathrm{Rem}(x, y) = 0\}$.

Definition 7.3. A partial function $\psi(x_1, \ldots, x_k)$ is said to be *partial recursive* if it is "computable" in the sense that there exists a program $\mathcal{P}$ with the foloowing property. For all $x_1, \ldots, x_k \in \mathbb{N}$, $\psi(x_1, \ldots, x_k)$ is defined if and only if $\mathcal{P}(x_1, \ldots, x_k)$ eventually halts, in which case the value $y = \psi(x_1, \ldots, x_k)$ appears in register R_{k+1}.

Remark 7.4. Given a program $\mathcal{P}$ and a positive integer $k \geq 1$, there is obviously a unique k-place partial recursive function $\psi :\subseteq \mathbb{N}^k \to \mathbb{N}$ computed by $\mathcal{P}$.

Note also that the k-place partial recursive functions which happen to be total are exactly what we have previously called the *computable* k-place functions (Definition 1.6).

Remark 7.5. One way that partial recursive functions arise naturally is from the μ-operator, as shown by the following lemma.

Lemma 7.6 (unbounded μ-operator). Given a $(k + 1)$-place computable predicate $R(y, x_1, \ldots, x_k)$, we have a k-place partial recursive function defined by

$$\begin{aligned}
\psi(x_1, \ldots, x_k) &\simeq \mu y\, R(y, x_1, \ldots, x_k) \\
&\simeq \begin{cases} \text{the least } y \text{ such that } R(y, x_1, \ldots, x_k) \text{ holds,} \\ \qquad \text{if such a } y \text{ exists,} \\ \text{undefined, if no such } y \text{ exists.} \end{cases}
\end{aligned}$$

Proof. See Exercise 1.11. □

Remark 7.7. Let $\psi(x_1, \ldots, x_k)$ be a k-place partial function. Using prime power coding, we can encode $\psi(x_1, \ldots, x_k)$ as a 1-place partial function, namely

$$\psi^*(w) \simeq \psi((w)_1, \ldots, (w)_k)$$

for all $w \in \mathbb{N}$. Clearly ψ is partial recursive if and only if ψ^* is partial recursive. Thus, for many purposes, the study of k-place partial recursive functions is equivalent to the study of 1-place partial recursive functions.

Definition 7.8 (Gödel numbering). To each register machine program $\mathcal{P}$ we shall assign a unique number $\#(\mathcal{P}) \in \mathbb{N}$, the *Gödel number* of $\mathcal{P}$. Thus we shall have a one-to-one function

$$\# : \{\text{programs}\} \xrightarrow{\text{1-1}} \mathbb{N}.$$

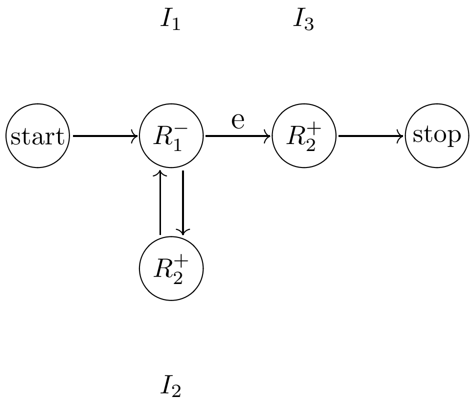
The idea is that $\#(\mathcal{P})$ will be a number which completely describes the program $\mathcal{P}$.

In order to define $\#(\mathcal{P})$, suppose $\mathcal{P}$ consists of l instructions of the form R_i^+ or R_i^- which are labeled $I_1, \ldots, I_l$. This list of instructions does not include the start and stop instructions. We always assume that I_1 is the instruction pointed to by the start instruction. We take I_0 to be the stop instruction. Using prime power coding, we define

$$\#(\mathcal{P}) = \prod_{m=1}^{l} p_m^{\#(I_m)}$$

where $\#(I_m)$, $m = 1, \ldots, l$ are numbers which completely describe the instructions I_m, $m = 1, \ldots, l$. These numbers are defined as follows. If I_m is an increment instruction R_i^+ pointing to I_n, let $\#(I_m) = 3^i \cdot 5^n$. If I_m is a decrement instruction R_i^- pointing to I_{n_0} if R_i is empty, I_{n_1} otherwise, let $\#(I_m) = 2 \cdot 3^i \cdot 5^{n_0} \cdot 7^{n_1}$. This completes the definition of $\#(\mathcal{P})$.

Example 7.9. Here is a program $\mathcal{P}$



with three instructions labeled I_1, I_2, I_3 not including the start and stop instructions. The Gödel number of $\mathcal{P}$ is found as follows:

$$\#(I_1) = 2 \cdot 3^1 \cdot 5^3 \cdot 7^2 = 6 \cdot 125 \cdot 49 = 36750$$

$$\#(I_2) = 3^2 \cdot 5^1 = 45$$

$$\#(I_3) = 3^2 \cdot 5^0 = 9$$

$$\#(\mathcal{P}) = p_1^{\#(I_1)} p_2^{\#(I_2)} p_3^{\#(I_3)} = 3^{36750} \cdot 5^{45} \cdot 7^9$$

Note that this number encodes the complete structure of the program $\mathcal{P}$.

Notation 7.10. For each $e, k \in \mathbb{N}$ we have a k-place partial recursive function $\varphi_e^{(k)}$ defined as follows. If $e = \#(\mathcal{P})$ for some program $\mathcal{P}$, then $\varphi_e^{(k)}$ is the k-place partial recursive function computed by $\mathcal{P}$. If e is not the Gödel number of a program, then $\varphi_e^{(k)}$ is the empty or totally undefined function.

In other words, for all nonnegative integers $e, k, x_1, \ldots, x_k$, the expression

$$\varphi_e^{(k)}(x_1, \ldots, x_k)$$

is defined as the content of R_{k+1} if and when $\mathcal{P}(x_1, \ldots, x_k)$ halts, provided e is the Gödel number of a program $\mathcal{P}$, and provided the run $\mathcal{P}(x_1, \ldots, x_k)$ eventually halts. Otherwise, $\varphi_e^{(k)}(x_1, \ldots, x_k)$ is undefined.

Example 7.11. Let $e = \#(\mathcal{P})$ as in Example 7.9. Note that $\mathcal{P}(x)$ simply adds 1 to x and outputs $x + 1$ in register R_2. Therefore, for this e and for all x, $\varphi_e^{(1)}(x)$ is defined and equal to $x + 1$.

Definition 7.12. An *index* of a partial recursive function is the Gödel number of a program which computes the function. In the previous example, e is an index of the 1-place function $x + 1$.

Remark 7.13. Clearly each partial recursive function has an index. Indeed, each partial recursive function has infinitely many indices, because there are infinitely many distinct programs which compute it.

8 The Enumeration Theorem

An important theorem on indices is the following.

Theorem 8.1 (the Enumeration Theorem). For each $k \geq 1$, the $(k+1)$-place function

$$\varphi_e^{(k)}(x_1, \ldots, x_k)$$

is partial recursive. Note that the arguments of this function are $e, x_1, \ldots, x_k$.

Remark 8.2. The Enumeration Theorem, due to Turing in 1936, embodies the idea of a "universal machine", i.e., a stored-program digital computer.

Proof of Theorem 8.1. The prooof uses a $(k+2)$-place function

$$\text{State}(e, x_1, \ldots, x_k, n) = p_0^m p_1^{z_1} \cdots p_s^{z_s} = z$$

which uses prime power coding to represent the state of $\mathcal{P}(x_1, \ldots, x_k)$ after n steps of computation. We write $e = \#(\mathcal{P})$ where $\mathcal{P}$ is a program with registers $R_1, \ldots, R_s$. Here I_m is the next instruction to be executed, and z_i is the content of register R_i. Note that $(z)_0 = m$ and $(z)_i = z_i$ for each $i = 1, \ldots, s$. Note also that this data together with $\mathcal{P}$ completely determines the run of $\mathcal{P}(x_1, \ldots, x_k)$ from step n onward.

The main point of the proof is that $\text{State}(e, x_1, \ldots, x_k, n)$ is a computable $(k+2)$-place function. This is proved by primitive recursion on n. We begin with

$$\text{State}(e, x_1, \ldots, x_k, 0) = p_0^1 p_1^{x_1} \cdots p_k^{x_k}$$

which means that at step 0 we are about to execute instruction I_1 and $x_1, \ldots, x_k$ are in registers $R_1, \ldots, R_k$. This is the initial state of the run $\mathcal{P}(x_1, \ldots, x_k)$. In general we have

$$\text{State}(e, x_1, \ldots, x_k, n+1) = \text{NextState}(e, \text{State}(e, x_1, \ldots, x_k, n))$$

where NextState is a function to be described. This means that the state of $\mathcal{P}(x_1, \ldots, x_k)$ after $n+1$ steps of computation is to be specified in terms of the program $\mathcal{P}$ and the state of $\mathcal{P}(x_1, \ldots, x_k)$ after n steps of computation. It remains to show that an appropriate 2-place function $\text{NextState}(e, z)$ is computable. The details of this are on page 29 of the 558 lecture notes, and below.

Note that $z = \text{State}(e, x_1, \ldots, x_k, n)$ is a halting state if and only if I_0 is the next instruction to be executed, i.e., $(z)_0 = 0$, in which case the output is $(z)_{k+1}$. Thus by Lemma 7.6 we have a partial recursive function

$$\text{StopTime}(e, x_1, \ldots, x_k) \simeq \mu n \, (\text{State}(e, x_1, \ldots, x_k, n))_0 = 0)$$

which gives the number of steps needed for $\mathcal{P}(x_1, \ldots, x_k)$ to halt, and then

$$\varphi_e^{(k)}(x_1, \ldots, x_k) \simeq (\text{State}(e, x_1, \ldots, x_k, \text{StopTime}(e, x_1, \ldots, x_k))_{k+1}$$

which completes the proof. □

Notation 8.3. Let E be an expression which may or may not be defined. We write $E \downarrow$ to mean that E is defined. We write $E \uparrow$ to mean that E is undefined. If E_1 and E_2 are two such expressions, we write $E_1 \simeq E_2$ to mean that E_1 and E_2 are both defined and have the same value, or both are undefined. The binary relation $\simeq$ is known as *strong equality.*

Lecture 5: September 6, 2007

Proof of Theorem 8.1, additional details. The heart of the proof is the function $\text{State}(e, x_1, \ldots, x_k, n)$. Recall that $e = \#(\mathcal{P})$ and $\mathcal{P}(x_1, \ldots, x_k)$ is the run of

$\mathcal{P}$ with inputs $x_1, \ldots, x_k$ in registers $R_1, \ldots, R_k$ and all other registers empty. The first instruction to be executed is I_1. After n steps of computation, we have $z_1, \ldots, z_s$ in registers $R_1, \ldots, R_s$ and we are about to execute instruction I_m for some m in the range $1 \leq m \leq l$. Then

$$\text{State}(e, x_1, \ldots, x_k, n) = z = p_0^m p_1^{z_1} \cdots p_s^{z_s}.$$

We use primitive recursion on n to show that the $(k+2)$-place function State is computable. Namely

$$\begin{aligned}
\text{State}(e, x_1, \ldots, x_k, 0) &= p_0^1 p_1^{x_1} \cdots p_k^{x_k} \\
\text{State}(e, x_1, \ldots, x_k, n+1) &= \text{NextState}(e, \text{State}(e, x_1, \ldots, x_k, n))
\end{aligned}$$

where the 2-place function NextState(e, z) does the following. It decodes from z all of the information about the state of $\mathcal{P}(x_1, \ldots, x_k)$ after n steps. It then decodes the Gödel number e to find the next instruction, and executes that instruction. It then recodes the state of $\mathcal{P}(x_1, \ldots, x_k)$ after $n+1$ steps, using prime power coding. In detail we have

$$\text{NextState}(e, z) = \begin{cases}
z \cdot p_i \cdot p_0^{-m+n_0} & \text{if } ((e)_m)_0 = 0 \text{ (increment)} \\[2ex]
z \cdot p_0^{-m+n_0} & \begin{array}{l}\text{if } ((e)_m)_0 = 1 \text{ and } (z)_i = 0 \\ \text{(decrement with } R_i \text{ empty)}\end{array} \\[2ex]
z \cdot p_i^{-1} \cdot p_0^{-m+n_1} & \begin{array}{l}\text{if } ((e)_m)_0 = 1 \text{ and } (z)_i > 0 \\ \text{(decrement with } R_i \text{ nonempty)}\end{array} \\[2ex]
z & \text{otherwise}
\end{cases}$$

where $m = (z)_0$ and $i = ((e)_m)_1$ and $n_0 = ((e)_m)_2$ and $n_1 = ((e)_m)_3$. Here we are using a method called *definition by cases*. Note that definition by cases is computable:

Lemma 8.4. If P, f_1, f_2 are computable, then

$$f(-) \quad = \quad \begin{cases} f_1(-) & \text{if } P(-) \text{ holds} \\ f_2(-) & \text{if } \neg\, P(-) \text{ holds} \end{cases}$$

is computable.

Proof. $f(-) = \chi_P(-) \cdot f_1(-) + \chi_{\neg\, P}(-) \cdot f_2(-)$. □

It is now clear that State$(e, x_1, \ldots, x_k, n)$ is computable as a $(k+2)$-place function of $e, x_1, \ldots, x_k, n$.

We shall also need the following easy lemma.

Lemma 8.5. The predicate

$$\text{Program}(e) \equiv \text{“}e \text{ is the Gödel number of a program”}$$

is computable.

Proof. This is straightforward, using bounded quantification. Roughly speaking, Program$(e) \equiv (\exists l < e)\,(e = \prod_{m=1}^{l} p_m^{(e)_m} \wedge (\forall m < e)\,(\text{if } 1 \leq m \leq l \text{ then } \exists i, j, k < e)\,((e)_m = 3^i \cdot 5^j \vee (e)_m = 2 \cdot 3^i \cdot 5^j \cdot 7^k))$, etc. □

We have seen that State$(e, x_1, \ldots, x_k, n) = z$ is a computable $(k+2)$-place function. Note that $(z)_0 = 0$ means $m = 0$, i.e., we are in a halting state, because I_0 is the stop instruction. Therefore, the function

$$\text{StopTime}(e, x_1, \ldots, x_k) \simeq \mu n\,((\text{State}(e, x_1, \ldots, x_k, n))_0 = 0 \wedge \text{Program}(e))$$

tells us how many steps it takes for $\mathcal{P}(x_1, \ldots, x_k)$ to halt. This is undefined if $\mathcal{P}(x_1, \ldots, x_k)$ never halts, or if e is not the Gödel number of a program. By Lemma 7.6 StopTime$(e, x_1, \ldots, x_k)$ is a $(k+1)$-place partial recursive function. Furthermore $(z)_{k+1}$ = the content of R_{k+1} after n steps. Therefore, we have

$$\varphi_e^{(k)}(x_1, \ldots, x_k) \simeq (\text{State}(e, x_1, \ldots, x_k, \text{StopTime}(e, x_1, \ldots, x_k)))_{k+1}$$

which shows that $\varphi_e^{(k)}(x_1, \ldots, x_k)$ is partial recursive as a function of $e, x_1, \ldots, x_k$. This completes the proof of the Enumeration Theorem. □

Remark 8.6. The Enumeration Theorem gives us a 2-place partial recursive function, namely $\varphi_e^{(1)}(x)$, which enumerates all 1-place partial recursive functions. Using this, we can exhibit a 1-place function which is not partial recursive. This is accomplished by means of a trick known as *diagonalization*, as in the proof of the following theorem and corollary.

Theorem 8.7. We can exhibit a 1-place partial recursive function which cannot be extended to a 1-place total recursive function.

Proof. It follows from Theorem 8.1 that the 1-place partial function

$$\psi(x) \quad \simeq \quad \varphi_x^{(1)}(x) + 1$$

is partial recursive. We claim that there is no extension of $\psi(x)$ to a total 1-place computable function $f : \mathbb{N} \to \mathbb{N}$. To see this, let $f(x)$ be a total 1-place function which extends $\psi(x)$, i.e., $f(x) = \psi(x)$ whenever $\psi(x)$ is defined. We claim that $f(x)$ is not computable. If $f(x)$ were computable, let e be an index of $f(x)$, i.e., $e = \#(\mathcal{P})$ where $\mathcal{P}$ is a program which computes $f(x)$. In other words, $f(x) = \varphi_e^{(1)}(x)$ for all x. In particular $\varphi_e^{(1)}(e) = f(e) \neq f(e) + 1 = \varphi_e^{(1)}(e) + 1$, a contradiction. This completes the proof. □

Corollary 8.8. We can exhibit a specific 1-place number-theoretic function which is not computable.

Proof. For example, the function

$$f(x) \quad = \quad \begin{cases} \varphi_x^{(1)}(x) + 1 & \text{if } \varphi_x^{(1)}(x) \text{ is defined} \\ 0 & \text{otherwise} \end{cases}$$

is noncomputable. □

Lecture 6: September 7, 2007

9 Comments on Homework #2

Here are some miscellaneous remarks and hints for Homework #2.

Computable real numbers

Recall from Exercise 3.6 that a positive real number r *computable* if and only if there exist computable sequences $a_n, b_n \in \mathbb{N}$, $n = 0, 1, 2, \ldots$, such that (1) the sequence of rational numbers a_n/b_n converges to r, and (2) $|r - a_n/b_n| < 1/2^n$ for all n. Note that condition (2) is important. Problem 1 cannot be solved without using this condition. The problem is to show that the real number r is computable if and only if the number-theoretic function $f(n) = n$th decimal digit of r is computable.

Recursive inseparability

Two sets $A, B \subseteq \mathbb{N}$ are said to be *recursively separable* if there exists a computable function f such that $f(n) = 0$ for all $n \in A$, $f(n) = 1$ for all $n \in B$, and $f(n) = 0$ or 1 otherwise. One of the homework problems is to show that the sets K_0 and K_1 (or more generally K_m and K_n for all $m \neq n$) are recursively inseparable. This will be accomplished by means of a diagonal argument.

Uncountable sets

By definition, the set $\mathbb{N}$ is countable, and any set that can be indexed by $\mathbb{N}$ is countable. For example, the set $\{\varphi_e^{(1)} \mid e \in \mathbb{N}\}$ is countable. In particular, there are only countable many computable number-theoretic functions.

On the other hand, a diagonal argument shows that various sets of number-theoretic functions are uncountable:

1. $\mathbb{N}^{\mathbb{N}} = \{f : \mathbb{N} \to \mathbb{N}\}$
2. $\{0,1\}^{\mathbb{N}} = \{f : \mathbb{N} \to \{0,1\}\}$
3. $S_\infty = \{\text{permutations of } \mathbb{N}\}$

Namely, given a countable sequence of functions $f_n : \mathbb{N} \to \mathbb{N}$, $n = 0, 1, 2, \ldots$, construct $g \notin \{f_n \mid n = 0, 1, 2, \ldots\}$ by letting $g(n) = f_n(n) + 1$ for all n. This shows that $\mathbb{N}^{\mathbb{N}}$ is uncountable. To show that $\{0,1\}^{\mathbb{N}}$ is uncountable, use a $0,1$-valued variant, namely

$$g(n) \quad = \quad \begin{cases} 1 & \text{if } f_n(n) = 0 \\ 0 & \text{if } f_n(n) \neq 0\,. \end{cases}$$

To show that S_∞ is uncoutable, define a permutation g by letting $g(2n) = 2n+1$ and $g(2n+1) = 2n$ if $f_n(n) = 2n$, otherwise $g(2n) = 2n$ and $g(2n+1) = 2n+1$.

In particular, we see that the group of recursive permutations of $\mathbb{N}$ is a countable subgroup of an uncountable group, namely the group of all permutations of $\mathbb{N}$. In other words, S_∞(recursive) is a countable subgroup of S_∞. See also Homework #2 Problem 3.

Note also that, if $g \in \{0,1\}^{\mathbb{N}}$ is noncomputable, then the real number $g(0).g(1)g(2)g(3)\cdots$ is noncomputable. Here $g(n)$ is the nth decimal digit of the number. See also Homework #2 Problem 1.

The above arguments show that the mentioned sets of functions are uncountable. It follows that they contain noncomputable functions. However, this kind of argument doesn't give us specific examples of noncomputable functions. Specific examples can be obtained as in Theorem 8.7 and 8.8 by means of the Enumeration Theorem.

The Parametrization Theorem

Recall that the Enumeration Theorem for 1-place partial recursive functions says:

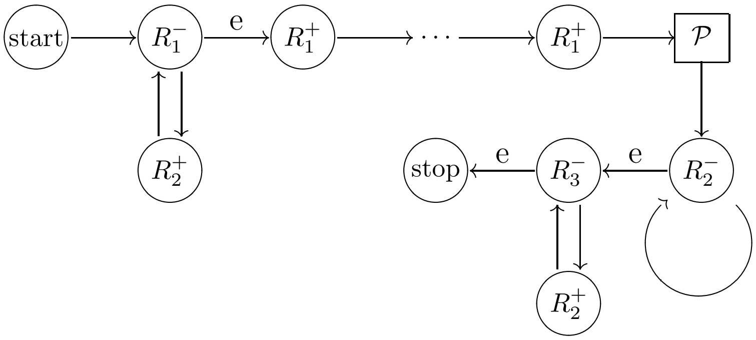
> $\varphi_e^{(1)}(x)$ is a 2-place partial recursive function (as a function of e, x).

An important supplement to the Enumeration Theorem is the Parametrization Theorem. The Parametrization Theorem for 1-place partial recursive functions says:

> Given a 2-place partial recursive function $\psi(w, x)$, we can find a 1-place total recursive function $h(w)$ such that
>
> $$\varphi_{h(w)}^{(1)}(x) \quad \simeq \quad \psi(w, x)$$
>
> for all w, x.

This theorem will be useful in solving Homework #2, Problem 6(a).

Proof of the Parametrization Theorem. Let $\mathcal{P}$ be a program which computes the 2-place partial recursive function $\psi(w, x)$. For each fixed w, consider the program



where the number of R_1^+ instructions is w. Call this program $\mathcal{Q}_w$.

To explain the operation of $\mathcal{Q}_w$, suppose we start $\mathcal{Q}_w$ with an arbitrary $x \in \mathbb{N}$ in R_1 and all other registers empty. We begin by transferring x to R_2. We then put w into R_1 by simply incrementing R_1 w times. We are now ready to run $\mathcal{P}(w, x)$. After running $\mathcal{P}(w, x)$, the output $\psi(w, x)$ is in R_3. We then clear R_2, transfer the output from R_3 to R_2, and halt.

Thus, starting with x in R_1, $\mathcal{Q}_w(x)$ eventually halts with $\psi(w, x)$ in R_2 provided $\psi(w, x)$ is defined. If $\psi(w, x)$ is undefined, $\mathcal{Q}_w(x)$ does not halt. In other words, $\mathcal{Q}_w$ is a program which computes $\psi(w, x)$ as a function of x. Defining $h(w) = \#(\mathcal{Q}_w)$, we see that $h(w)$ has the desired property. It remains to show that $h(w)$ is computable as a function of w, but this is straightforward. □

An example of an unsolvable problem

Let $A \subseteq \mathbb{N}$. We associate to A what is called a *decision problem*, namely:

Given n, to "decide" whether $n \in A$ or not.

This decision problem is said to be *solvable* if A is computable, i.e., $\chi_A(n)$ is a recursive 1-place function. Otherwise, the decision problem is said to be *unsolvable*.

Example 9.1. Consider the set $K \subseteq \mathbb{N}$ defined by

$$K = \{x \in \mathbb{N} \mid \varphi_x^{(1)}(x) \downarrow\}.$$

We claim that K is noncomputable, i.e., the decision problem for K is unsolvable.

To see this, suppose K were recursive (i.e., computable). Consider the function

$$f(x) \quad = \quad \begin{cases} \varphi_x^{(1)}(x) + 1 & \text{if } x \in K \\ 0 & \text{if } x \notin K\,. \end{cases}$$

If K were computable, $f(x)$ would be computable, using definition by cases plus the Enumeration Theorem. But we already know (by diagonalization, see Corollary 8.8) that this particular function $f(x)$ is noncomputable.

Remark 9.2. The decision problem for K is our first example of an unsolvable mathematical problem. Later we shall see examples of unsolvable decision problems from various branches of mathematics.

Note also that $K = \bigcup_{n=0}^{\infty} K_n$ where K_n is as in Homework #2, Problem 5.

Diagonal nonrecursiveness

Definition 9.3. A 1-place total number-theoretic function $g : \mathbb{N} \to \mathbb{N}$ is said to be *diagonally non-recursive* (abbreviated DNR) if

$$g(x) \quad \not\simeq \quad \varphi_x^{(1)}(x)$$

for all x.

Remark 9.4. Instead of $\varphi_x^{(1)}(x)$ we could use any universal partial recursive function, as explained in Homework #2, Problems 6 and 7.

Obviously, any DNR function is nonrecursive. A possible research project for this course is to study DNR functions and their Turing degrees.

10 Homework #3, due September 17, 2007

Exercises 10.1. Recall that $W_x = \mathrm{dom}(\varphi_x^{(1)})$. Note that W_x, $x = 0, 1, 2, \ldots$, is the standard recursive enumeration of the recursively enumerable subsets of $\mathbb{N}$.

1. Which many-one reducibility relations hold or do not hold among the following sets and their complements?

   $$K = \{x \mid x \in W_x\}$$
   $$H = \{x \mid 0 \in W_x\}$$
   $$T = \{x \mid W_x = \mathbb{N}\}$$
   $$E = \{x \mid W_x = \emptyset\}$$
   $$S = \{x \mid W_x \text{ is infinite}\}$$

   Prove your answers.

   Hint: Each of these sets is many-one complete within an appropriate level of the arithmetical hierarchy.

2. A set $P \subseteq \mathbb{N}$ is said to be *productive* if there exists a total recursive function $h(x)$ such that for all x, if $W_x \subseteq P$ then $h(x) \notin W_x$ and $h(x) \in P$. Such a function is called a *productive function* for P.

   A *creative set* is a recursively enumerable set whose complement is productive.

   Prove the following.

   (a) K is creative.

   (b) If A and B are recursively enumerable sets and $A \leq_m B$ and A is creative, then B is creative.

   (c) If B is recursively enumerable and many-one complete, then B is creative.

   (d) (Extra Credit) If B is creative, then B is many-one complete.

   (e) (Extra Credit) If A and B are creative, then A and B are *recursively isomorphic*. This means that there exists a recursive permutation of $\mathbb{N}$, call it g, such that $x \in A$ if and only if $g(x) \in B$, for all x.

3. A set $I \subseteq \mathbb{N}$ is said to be *immune* if I is infinite yet includes no infinite recursively enumerable set.

   A *simple set* is a recursively enumerable set whose complement is immune.

   Prove the following.

   (a) If A is simple, then A is not recursive.

   (b) If A is simple, then A is not creative.

4. Let $f : \mathbb{N} \xrightarrow{1-1} \mathbb{N}$ be a one-to-one total recursive function such that the range of f is nonrecursive. The *deficiency set* of f is defined as

$$D_f = \{x \mid \exists y \, (x < y \wedge f(x) > f(y))\} .$$

   Prove that D_f is a simple set.

   Conclude that there exist recursively enumerable sets which are neither recursive nor many-one complete.

5. (Extra Credit) Generalize Exercises 2, 3, 4 to higher levels of the arithmetical hierarchy. Conclude that for each $n \geq 1$ there exist Σ_n^0 sets which are neither Δ_n^0 nor many-one complete within the class of Σ_n^0 sets.

Lecture 7: September 10, 2007

11 Unsolvable problems in core mathematics

We have defined a *decision problem* to be a set $A \subseteq \mathbb{N}$. The problem A is said to be *solvable* if A is recursive, i.e., computable. As an example of an unsolvable problem, we have mentioned $K = \{x \mid \varphi_x^{(1)}(x) \downarrow\}$.

We shall now discuss two examples of unsolvable problems which are interesting from the viewpoint of core mathematics:

1. Hilbert's 10th Problem.

2. The Word Problem for Groups.

Remark 11.1. In the Pennsylvania State University Department of Mathematics, there are several experts on these topics. Professor Kirsten Eisentraeger, who joined the department recently, is an expert on generalizations of Hilbert's 10th Problem. Professor Alexandra Shlapentokh is another expert on Hilbert's 10th problem who will visit the department and give a MASS colloquium on November 8, 2007. Professor Alexander Nabutovsky is an expert on unsolvability of the Word Problem for Groups and its applications to geometry.

Hilbert's 10th Problem

In a famous talk in 1900 Hilbert listed 23 problems which were intended to set the tone for 20th century mathematics. Included in the list were several problems in mathematical logic and foundations of mathematics. Problem 1 was the continuum hypothesis, while Problem 2 concerned the formal consistency of mathematics. Problem 10 on Hilbert's list read as follows:

> Given a Diophantine equation $f(x_1, \ldots, x_k) = 0$, does this equation have a solution in integers $x_1, \ldots, x_k$? The problem is to construct an algorithm for answering this question given the polynomial f.

(By definition, a *Diophantine equation* is a polynomial equation with integer coefficients.)

Theorem 11.2 (Matiyasevich 1969, also Davis/Putnam/Robinson 1950s). Hilbert's 10th Problem is unsolvable.

In order to state Matiyasevich's Theorem rigorously, we use Gödel numbering to translate Hilbert's 10th Problem into a "decision problem" as above. That is, we define a function

$$\# : \{\text{Diophantine equations}\} \xrightarrow{1-1} \mathbb{N}\,.$$

Let E be a Diophantine equation

$$f(x_1, \ldots, x_k) = \sum_{e_1,\ldots,e_k} a_{e_1 \cdots e_k} x_1^{e_1} \cdots x_k^{e_k} \quad = \quad 0$$

where $e_1, \ldots, e_k \in \mathbb{N}$ and $a_{e_1 \cdots e_k} \in \mathbb{Z}$. To define $\#(E)$, we first Gödel number the integers

$$\begin{aligned} \#(a) &= 2a && \text{for } a \geq 0 \\ \#(-a) &= 2a - 1 && \text{for } a > 0 \end{aligned}$$

and the k-tuples

$$\#(\langle e_1, \ldots, e_k \rangle) = p_1^{e_1} \cdots p_n^{e_k}$$

(prime power coding). Then we define

$$\#(E) = \prod_{e_1,\ldots,e_k} p_{\#(\langle e_1,\ldots,e_k \rangle)}^{\#(a_{e_1 \cdots e_k})}$$

and this allows us to state Matiyasevich's Theorem precisely:

Theorem 11.3 (Matiyasevich 1969). The set of Gödel numbers

$$\{\#(E) \mid E \text{ is a Diophantine equation, } E \text{ has a solution in integers}\}$$

is nonrecursive.

Proof. The proof is long and involves some non-trivial number theory. Full details are in my Spring 2005 lecture notes, available on the course web page. □

The word problem for groups

Consider a finite set of symbols $A = \{a_1, \ldots, a_k\}$. By a *relation* we mean a group-theoretic equation on these symbols. Such an equation may be written in the form $W = 1$ where W is a *word*, i.e., a concatenation of symbols chosen from $a_1, \ldots, a_k, a_1^{-1}, \ldots, a_k^{-1}$. Given a finite set of relations

$$R = \{W_1 = 1, \ldots, W_l = 1\}$$

there is a unique largest group $G = \langle A \mid R \rangle$ generated by $a_1, \ldots, a_k$ satisfying the given set of relations R.

Example 11.4. Consider the generators $A = \{a, b\}$ and the relations

$$R = \{ab = ba, a^2 = 1, b^3 = 1\}.$$

Note also that the relation $ab = ba$ can be written using a word as $aba^{-1}b^{-1} = 1$. We can use the relations to simplify words on this alphabet. For example

$$aba^2ba^{-1}b^{-1}a^{-1}b^2a^2ba^{-1} = a^2b^4 = b.$$

The group $G = \langle A \mid R \rangle$ defined by these generators and relations is isomorphic to $C_2 \times C_3$ with elements $\{1, a, b, b^2, ab, ab^2\}$. Here C_n is the cyclic group of order n.

Remark 11.5. Finitely presented groups arise naturally in algebraic topology and geometry. For instance, the fundamental group of a finite simplicial complex is a finitely presented group.

For each finitely presented group $G = \langle A \mid R \rangle$, the *word problem for G* is the following problem:

> Given a word W on the finite set of generators A, does the equation $W = 1$ hold in G? The problem is to construct an algorithm for deciding whether $W = 1$ in G.

This problem was originally posed by the group theorist Dehn in the 1890s.

Note that for G as in Example 11.4, the word problem is solvable, because there is an algorithm to simplify each word and thus identify it with one of the six elements of G.

Theorem 11.6 (Boone, Novikov, 1950's). We can construct a finitely presented group $G = \langle A \mid R \rangle$ such that the word problem for G is unsolvable.

In order to state this theorem rigorously, we define a Gödel numbering

$$\# : \{\text{words on } a_1, \ldots, a_k\} \xrightarrow{1-1} \mathbb{N}\,.$$

We first Gödel number the symbols $a_1, \ldots, a_k, a_1^{-1}, \ldots, a_k^{-1}$ by

$$\begin{aligned} \#(a_i) &= 2i \\ \#(a_i^{-1}) &= 2i + 1 \end{aligned}$$

and then words $W = a_{i_1}^{\epsilon_1} \cdots a_{i_n}^{\epsilon_n}$ can be Gödel numbered as

$$\#(a_{i_1}^{\epsilon_1} \cdots a_{i_n}^{\epsilon_n}) = \prod_{j=1}^{n} p_j^{\#(a_{i_j}^{\epsilon_j})}$$

where $i_j \in \{1, \ldots, k\}$ and $\epsilon_j = \pm 1$. We can then state:

Theorem 11.7 (Boone, Novikov, 1950's). We can construct a finitely presented group $G = \langle A \mid R \rangle$ such that the set of Gödel numbers

$$\{\#(W) \mid W \text{ is a word on } A,\, W = 1 \text{ in } G\}$$

is nonrecursive.

Proof. The proof is long and involves some non-trivial group theory. Full details are in my Spring 2005 lecture notes, available on the course web page. □

12 Techniques for classifying unsolvable problems

Let $A, B \subseteq \mathbb{N}$ be decision problems.

Definition 12.1. A is *many-one reducible to* B, abbreviated $A \leq_m B$, if there exists a total recursive function $h(x)$ such that for all x, $x \in A$ if and only if $h(x) \in B$.

Intuitively this means that, if we knew how to solve the problem B, we could then solve the problem A as follows. Given x, to decide whether $x \in A$, compute $h(x)$ and ask whether $h(x) \in B$. If $h(x) \in B$, then $x \in A$. Conversely, if $h(x) \notin B$, then $x \notin A$.

Supporting this intuition, we have the following lemma.

Lemma 12.2. If $A \leq_m B$ and B is recursive ("solvable"), then A is recursive ("solvable").

Proof. $\chi_A(x) = \chi_B(h(x))$. □

Lemma 12.3. If $A \leq_m B$ and A is nonrecursive, then B is nonrecursive.

Proof. This is just the contrapositive of the previous lemma. □

Corollary 12.4. If $K \leq_m B$, then B is nonrecursive.

Proof. Obvious, because K is nonrecursive. □

Remark 12.5. The previous corollary can be used to demonstrate the unsolvability of many mathematical problems, including Hilbert's 10th Problem and the Word Problem for Groups. One shows that the unsolvable problem K is reducible to each of these problems, hence they too must be unsolvable.

1. In the case of Hilbert's 10th Problem, one constructs a specific Diophantine equation $f(x_0, x_1, \ldots, x_k) = 0$ such that for all $n \in \mathbb{N}$, $n \in K$ if and only if there exist integers $x_1, \ldots, x_k$ such that $f(n, x_1, \ldots, x_k) = 0$. Thus the recursive function

   $$n \mapsto \#(f(n, x_1, \ldots, x_k) = 0)$$

   shows that K is many-one reducible to Hilbert's 10th Problem.

   Note: By working harder, one can find an equation

   $$f(x, x_1, \ldots, x_k) = 0$$

   as above with $k = 9$. An open question is, what is the smallest number of variables, k, which will suffice. It is known that $k = 2$ will not suffice.

2. In the case of the Word Problem for groups, the rough idea is to construct a specific, finitely presented group G with generators a, b, c, d and some other generators having the following property: for all $n \in \mathbb{N}$, $n \in K$ if and only if $ab^n cb^{-n} d = 1$ in G. Thus the unsolvable problem K is many-one reducible to the word problem for this particular group G.

   Note: By working harder, one can construct a finitely presented group G with only two generators such that K is many-one reducible to the word problem for G.

Remark 12.6. Conversely, it is also true (and relatively easy to prove, using the Parametrization Theorem) that Hilbert's 10th Problem and the Word Problem for Groups are many-one reducible to K. Thus, all of these unsolvable problems are equivalent to each other, in the sense that they all have the same "degree of unsolvability." Later we shall define and study degrees of unsolvability in detail.

Lecture 8: September 12, 2007

13 The arithmetical hierarchy

To motivate the arithmetical hierarchy, consider K, our favorite example of a non-recursive set. We have

$$\begin{aligned} x \in K &\equiv \varphi_x^{(1)}(x)\downarrow \\ &\equiv \underbrace{\exists y \quad \underbrace{\left[\underbrace{(\text{State}(x,x,y))_0 = 0}_{\text{a halting state}}\right]}_{\text{a recursive predicate } R(x,y)}}_{\text{a } \Sigma^0_1 \text{ predicate}} \end{aligned}$$

So, although K is not recursive, K is described by a recursive predicate plus one existential quantifier, $\exists y$.

We shall see that Σ^0_1 is "level 1" of the arithmetical hierarchy. Generally speaking, the arithmetical hierarchy is a method of classifying non-recursive predicates according to the number of quantifiers needed to describe them.

Definition 13.1. As usual, abbreviate $x_1, \ldots, x_k$ as $-$.

1. A k-place predicate $P(-)$ is said to be Σ^0_n ($n \geq 1$) if $P(-)$ can be written in the form

   $$P(-) \equiv \underbrace{\exists y_1\ \forall y_2\ \exists y_3\ \forall y_4\ \cdots\ \begin{matrix}\exists\\ \forall\end{matrix}\ y_n}_{n \text{ alternating quantifiers starting with } \exists}\ R(-, y_1, y_2, \ldots, y_n)$$

   where R is a recursive predicate.

2. A k-place predicate $P(-)$ is Π^0_n ($n \geq 1$) if $P(-)$ can be written in the form

   $$P(-) \equiv \underbrace{\forall y_1\ \exists y_2\ \forall y_3\ \exists y_4\ \cdots\ \begin{matrix}\forall\\ \exists\end{matrix}\ y_n}_{n \text{ alternating quantifiers starting with } \forall}\ R(-, y_1, y_2, \ldots, y_n)$$

   where R is a recursive predicate.

Example 13.2. Consider the 1-place predicate (i.e., the set) $T \subseteq \mathbb{N}$ defined by

$$e \in T \ \equiv\ \varphi^{(1)}_e \text{ is a total function.}$$

To classify T in the arithmetical hierarchy, we have

$$\begin{aligned} e \in T &\equiv \forall x\, \varphi^{(1)}_e(x) \downarrow \\ &\equiv \forall x\, \exists y\, [\, (\mathrm{State}(e, x, y))_0 = 0\,] \\ &\equiv \Pi^0_2 \end{aligned}$$

so T belongs to the class Π^0_2. Question: does T belong to the class Σ^0_2? Later we shall develop a method for answering such questions.

Exercise 13.3. Prove that T is not recursive. (Hint: Use the Parametrization Theorem.)

The arithmetical hierarchy has some useful closure properties, expressed in the next four propositions.

Proposition 13.4. Each of the classes Σ^0_n and Π^0_n is closed under substitution of total recursive functions.

This means that, if $P(-, y)$ is a Σ^0_n predicate and $f(-)$ is a total recursive function, then the predicate

$$Q(-) \equiv P(-, f(-))$$

is again Σ^0_n. (And similarly for Π^0_n.)

Proof. For example, suppose P is Σ^0_3, i.e., $P(-,y) \equiv \exists u\, \forall v\, \exists w\, R(-,y,u,v,w)$ where $R(-,y,u,v,w)$ is recursive. Then $Q(-) \equiv \exists u\, \forall v\, \exists w\, R(-,f(-),u,v,w)$ so Q is again Σ^0_3. □

Proposition 13.5. The classes Σ^0_n and Π^0_n for $n \geq 1$ are closed under $\wedge$, $\vee$, $\forall x < y$, $\exists x < y$.

Caution: The classes Σ^0_n and Π^0_n for $n \geq 1$ are not closed under negation. For instance, K is Σ^0_1 but not Π^0_1, and $\neg K$ is Π^0_1 but not Σ^0_1.

Proof. We have

$$\begin{aligned}
\exists y\, P(-,y) \wedge \exists y\, Q(-,y) &\equiv \exists y\, \exists z\, [\, P(-,y) \wedge Q(-,z)\,] \\
&\equiv \exists w\, [\, P(-,(w)_1) \wedge Q(-,(w)_2)\,]\ .
\end{aligned}$$

Thus consecutive $\exists$'s (and similarly, consecutive $\forall$'s) can be reduced to a single $\exists$ (or a single $\forall$) by means of a pairing function or prime power coding.

Bounded quantifiers can be handled by means of Lemma 4.4 plus the following manipulation:

$$(\forall x < y)\, \exists z\, P(x,y,z,-) \quad \equiv \quad \exists w\, (\forall x < y)\, (\exists z < w)\, P(x,y,z,-)\,.$$

□

Proposition 13.6. Σ^0_n is closed under $\exists$. Π^0_n is closed under $\forall$.

Proof. Suppose $P(-,y)$ is Σ^0_n.

$$\begin{aligned}
Q(-) &\equiv \exists y\, P(-,y) \\
&\equiv \exists y\, \exists y_1\, \forall y_2\, \cdots y_n\, R(-,y,y_1,\ldots,y_n) \\
&\equiv \exists w\, \forall y_2\, \cdots y_n\, R(-,(w)_1,(w)_2,\ldots,y_n) \\
&\equiv \Sigma^0_n
\end{aligned}$$

□

The next example illustrates how our closure properties can be useful in classifying unsolvable problems.

Example 13.7. Let $S = \{e \mid \text{the domain of } \varphi^{(1)}_e \text{ is infinite}\}$. Classifying S in the arithmetical hierarchy, we have

$$\begin{aligned}
e \in S &\equiv \varphi_e^{(1)}(x)\downarrow \text{ for infinitely many } x \\
&\equiv \forall y\ \exists x\ (\ x > y \wedge \varphi_e^{(1)}(x)\downarrow) \\
&\equiv \underbrace{\forall y\ \underbrace{\exists x\ \underbrace{(\ \underbrace{x > y}_{\text{recursive}} \wedge \underbrace{\exists s\ \underbrace{(\mathrm{State}(e,x,s))_0 = 0}_{\text{recursive}}}_{\Sigma^0_1}\ )}_{\Sigma^0_1}}_{\Sigma^0_1}}_{\Pi^0_2}
\end{aligned}$$

Thus we see that S is Π^0_2. This kind of computation is called a Tarski/Kuratowski computation.

Proposition 13.8. If $P(-)$ is Σ^0_n, then $\neg P(-)$ is Π^0_n. If $P(-)$ is Π^0_n, then $\neg P(-)$ is Σ^0_n.

Proof. $\neg\exists \equiv \forall\neg$ and $\neg\forall \equiv \exists\neg$

$\neg\exists y_1\,\forall y_2\,\exists y_3 \equiv \forall y_1\,\neg\forall y_2\,\exists y_3 \equiv \forall y_1\,\exists y_2\,\neg\exists y_3 \equiv \forall y_1\,\exists y_2\,\forall y_3\,\neg$ □

Notation 13.9. W_e = domain of $\varphi_e^{(1)} = \{x \mid \varphi_e^{(1)}(x)\downarrow\}$.
Here e is called an *index* of the set W_e.

Note that some of our previous examples can be expressed concisely using this notation. In particular we have $K = \{e \mid e \in W_e\}$, $T = \{e \mid W_e = \mathbb{N}\}$, $S = \{e \mid W_e \text{ is infinite}\}$. Another such example is:

Example 13.10. $R = \{e \mid W_e \text{ is recursive}\}$.

Exercise 13.11. Use a Tarski/Kuratowski computation to classify R in the arithmetical hierarchy.

(picture of arithmetical hierarchy)

Lecture 9: September 13, 2007

We continue our discussion of the arithmetical hierarchy. The following theorem is a useful characterization of the lowest level of the hierarchy, Σ^0_1.

As usual, we denote $x_1, \ldots, x_k$ by $-$.

Theorem 13.12. $P \subseteq \mathbb{N}^k$, a k-place predicate, is Σ^0_1 if and only if P is the domain of some partial recursive function.

Proof. (⇐):
Assume $P = \mathrm{dom}(\varphi_e^{(k)})$

$$\begin{aligned} \text{Then } P(-) &\equiv \varphi_e^{(k)}\downarrow \\ &\equiv \underbrace{\exists n\ \underbrace{(\mathrm{State}(e,-,n))_0 = 0}_{\text{recursive}}}_{\Sigma_1^0} \end{aligned}$$

(⇒):
$P(-)$ is Σ_1^0.
Then $P(-) \equiv \exists y\, R(-,y)$ where R is recursive. Then $\psi(-) \simeq \mu y R(-,y)$ is a partial recursive function, and $P = \mathrm{dom}(\psi)$ □

In order to distinguish the levels of the arithmetical hierarchy, we first introduce universal predicates at each level.

Definition 13.13. A $(k+1)$-place predicate $U(e,-)$ is *universal* Σ_n^0 (for k-place predicates) if

1. $U(e,-)$ is Σ_n^0
2. every k-place Σ_n^0 predicate is $\equiv U(e,-)$ for some fixed e.

(Universal Π_n^0 predicates are similarly defined.)

Theorem 13.14. For each $k \geq 1$, $n \geq 1$, there exist universal Σ_n^0 and Π_n^0 predicates.

Proof.

1. $U(e,-) \equiv \varphi_e^{(k)}(-)\downarrow$ is universal Σ_1^0.
2. If $U(e,-,y)$ is univ Σ_n^0 (for $(k+1)$-place predicates), then $\forall y\, U(e,-,y)$ is universal Π_{n+1}^0 (for k-place predicates).
3. If $U(e,-)$ is universal Σ_n^0, then $\neg\, U(e,-)$ is universal Π_n^0; and if $U(e,-)$ is universal Π_n^0, then $\neg\, U(e,-)$ is universal Σ_n^0

□

Theorem 13.15. For each n, there exist sets which are Σ_n^0 but not Π_n^0 (or Π_n^0 but not Σ_n^0).

Proof. Let $U_n(e,x)$ be a universal Σ_n^0 predicate. Let $K_n \equiv \{e \mid U_n(e,e) holds\}$. Clearly K_n is Σ_n^0. The usual diagonal argument shows that K_n is not Π_n^0. By taking the complement, we obtain a set which is Π_n^0 and not Σ_n^0. □

Definition 13.16. $\Delta_n^0 = \Sigma_n^0 \cap \Pi_n^0$

Theorem 13.17. $P(-)$ is Δ_1^0 if and only if $P(-)$ is recursive.

Proof. (⇐):
trivial

(⇒):
$P(-)$ is Δ_1^0, so $P(-) \equiv \forall y\, R_1(-,y)$ and $P(-) \equiv\ \exists z\, R_2(-,z)$. □

Exercise 13.18. Show that there exist sets which are Δ_{n+1}^0 and neither Σ_n^0 nor Π_n^0.

14 Many-one reducibility and the arithmetical hierarchy

Recall that, for $A, B \subseteq \mathbb{N}$, $A \leq_m B$ means "$\exists$ total recursive function $h(x)$ such that $x \in A \equiv h(x) \in B$ for all x.

Easy Facts:

1. $A \leq_m A$ (reflexive)
2. $A \leq_m B$ and $B \leq_m C$ implies $A \leq_m C$ (transitivity)

   (via the function $f(g(x))$)
3. $A \oplus B =$ least upper bound of A, B under $\leq_m$.

   where $A \oplus B = \{2n \mid n \in A\} \cup \{2n+1 \mid n \in B\}$

   $A \leq_m A \oplus B$ via $n \mapsto 2n$ and $B \leq_m A \oplus B$ via $n \mapsto 2n+1$

   If $A \leq_m C$, $B \leq_m C$, then $A \oplus B \leq_m C$.
4. If $A \leq_m B$ and B is Σ^0_n, then A is Σ^0_n (similarly for Π^0_n and Δ^0_n).

   Pf: Because Σ^0_n is closed under recursive substitution.
5. If A, B are Σ^0_n, $A \oplus B$ is Σ^0_n.

Definition 14.1. A set $C \subseteq \mathbb{N}$ is called Σ^0_n *complete* if

1. C is Σ^0_n
2. Every Σ^0_n set is $\leq_m C$.

(Similarly for Π^0_n.)

Theorem 14.2. 1. Σ^0_n complete sets exist (for each $n \geq 1$)

2. Any such set is not Π^0_n.

Proof. 1. Let $U_n(e, x)$ be a universal Σ^0_n predicate. Let $C_n = \{3^e 5^x \mid U_n(e, x) \text{ holds}\}$. This is Σ^0_n complete.

2. If C is any Σ^0_n complete set, we have in particular $K_n \leq_m C$. Since K_n is not Π^0_n, C is not Π^0_n.

□

Example 14.3. $T = \{e \mid \varphi^{(1)}_e \text{ is total}\}$

Claim: T is Π^0_2 complete. (Hence T is not Σ^0_2.)

Proof.
$$\begin{aligned} e \in T &\equiv \varphi^{(1)}_e \text{ is total} \\ &\equiv \forall x\, \varphi^{(1)}_e(x) \downarrow \end{aligned}$$

(Σ^0_1)

(Π^0_2)

Suppose A is any Π^0_2 set. We need to show $A \leq_m T$. We have $x \in A \equiv \forall y\, \exists z\, R(x,y,z)$ where R is recursive. We then have a partial recursive function $\psi(x,y) \simeq \mu z\, R(x,y,z)$. By the Parametrization Theorem, there is a total recursive function $h(x)$ such that $\varphi^{(1)}_{h(x)}(y) \simeq \psi(x,y)$ for all x, y. Then

$$\begin{aligned} x \in A &\equiv \forall y\, \exists z\, R(x,y,z) \\ &\equiv \forall y\, \psi(x,y)\downarrow \\ &\equiv \forall y\, \varphi^{(1)}_{h(x)}(y)\downarrow \\ &\equiv \varphi^{(1)}_{h(x)} \text{ is total} \end{aligned}$$

□

15 Recursively enumerable sets

Recall the sets $K = \{x \mid \varphi^{(1)}_x(x)\downarrow\}$ and $H = \{x \mid \varphi^{(1)}_x(0)\downarrow\} = \{\#(\mathcal{P}) \mid \mathcal{P} \text{ halts}\} =$ the Halting Problem.

K, H are Σ^0_1 sets. We claim they are Σ^0_1 complete. To see this, let A be any Σ^0_1 set. To show $A \leq_m K$, $A \leq_m H$, note that $x \in A \equiv \exists y\, R(x,y)$ where $R(x,y)$ is a recursive predicate. We have a partial recursive function $\psi(x,z) \simeq \mu y\, R(x,y)$. By the Parametrization Theorem, we get a recursive function $h(x)$ such that $\varphi^{(1)}_{h(x)}(z) \simeq \psi(x,z)$. Then

$$\begin{aligned} x \in A &\equiv \exists y\, R(x,y) \\ &\equiv \psi(x,z)\downarrow \\ &\equiv \varphi^{(1)}_{h(x)}(z)\downarrow \ \text{(independent of } z\text{)} \\ &\equiv \varphi^{(1)}_{h(x)}(0)\downarrow \equiv x \in H \\ &\equiv \varphi^{(1)}_{h(x)}(h(x))\downarrow \equiv x \in K \end{aligned}$$

Exercise 15.1. Show that K_n is Σ^0_n complete.

Remark 15.2. K, H, S, T, R, E are all complete at appropriate levels of the arithmetical hierarchy.

But, there exist Σ^0_n and Π^0_n sets which are not complete.

Remark 15.3. Hilbert's 10th Problem, the Word Problem for Groups, etc. are Σ^0_1 complete. Thus, they are equivalent to the Halting Problem.

Theorem 15.4 (Recursively Enumerable Sets). For $A \subseteq \mathbb{N}$, t.f.a.p.e. (the following are pairwise equivalent):

1. A is Σ^0_1
2. $A =$ domain of a partial recursive function
3. $A = W_e = \operatorname{dom}(\varphi^{(1)}_e)$ for some e
4. $A =$ range of a partial recursive function
5. $A = \emptyset$ or A is the range of a total recursive function
6. A is finite or A is the range of a 1-1 total recursive function

We call such sets A *recursively enumerable* sets, or r.e. sets for short.

Proof. $1 \Leftrightarrow 2 \Leftrightarrow 3$ is obvious, and $6 \Rightarrow 5 \Rightarrow 4$ is obvious.

$4 \Rightarrow 1$:

Suppose $A =$ range of $\varphi_e^{(1)} = \{y \mid \exists x\, \varphi_e^{(1)}(x) \simeq y\}$.

$$\begin{aligned} \text{Then: } y \in A &\equiv \exists x\, \varphi_e^{(1)}(x) \simeq y \\ &\equiv \underbrace{\exists x\, \exists n \left[\underbrace{(\text{State}(e,x,n))_0 = 0 \wedge (\text{State}(e,x,n))_2 = y}_{\text{recursive}}\right]}_{\Sigma^0_1} \\ &\equiv \Sigma^0_1 \end{aligned}$$

We will show $1 \Rightarrow 6$ to complete the proof, but first a lemma:

Lemma 15.5. Let $B \subseteq \mathbb{N}$ be an infinite recursive set. Then the *principal function*

$$\pi_B(n) = n^{\text{th}} \text{ element of } B \text{ in increasing order}$$

is a total recursive function.

$$\begin{aligned} \text{Pf:}\quad \pi_B(0) &= \mu w\,(w \in B) \\ \pi_B(n+1) &= \mu w\,(w \in B \wedge w > \pi_B(n)) \end{aligned}$$

Now, to prove $1 \Rightarrow 6$:

Assume A is infinite and Σ^0_1, $A = \{x \mid \exists y\, R(x,y)\}$ where $R(x,y)$ is a recursive predicate. Let $\psi(x) \simeq \mu y\, R(x,y)$ (a partial recursive function) and $B = \{3^x 5^{\psi(x)} \mid x \in A\} = \{3^x 5^y \mid \underbrace{R(x,y) \wedge \neg \exists z < y\, R(x,z)}_{\text{recursive}}\}$.

Note that B is infinite and recursive, so π_B is recursive. Then $B =$ the range of π_B, and π_B is one-to-one. Define $f(n) = (\pi_B(n))_1$. Then $A =$ the range of f; and by construction, f is one-to-one and recursive. □

Remark 15.6. Define a set $A \subseteq \mathbb{N}$ to be *Diophantine* if there exists a polynomial $f(w, x_1, \ldots, x_k)$ with integer coefficients such that

$$A = \{n \mid \exists x_1 \cdots \exists x_k\, f(n, x_1, \ldots, x_k) = 0\}.$$

Here n ranges over $\mathbb{N}$ and $x_1, \ldots, x_k$ range over $\mathbb{Z}$. Obviously every Diophantine set is Σ^0_1. It follows by the previous theorem that every Diophantine set is recursively enumerable. A consequence of Matiyasevich's solution of Hilbert's 10th Problem is:

> A is Diophantine if and only if A is recursively enumerable.

This is another nice characterization of r.e. sets.

16 Homework #4, due September 24, 2007

Exercises 16.1. In this set of exercises we explore the structure of the Turing degrees. Given two Turing degrees $\mathbf{a}$ and $\mathbf{b}$, we know that the least upper bound $\sup(\mathbf{a}, \mathbf{b})$ always exists. Exercises 2, 4, and 7 below show that the greatest lower bound $\inf(\mathbf{a}, \mathbf{b})$ sometimes exists and sometimes does not exist.

For any Turing oracle f we have

$$f' = H^f = \{x \mid \varphi_x^{(1),f}(0)\downarrow\} = \text{the Halting Problem relative to } f.$$

We know that f' is a complete Σ_1^0 set relative to the oracle f. For any Turing degree $\mathbf{a} = \deg_T(f)$ we define

$$\mathbf{a}' = \deg_T(f') = \text{the Turing jump of } \mathbf{a}.$$

Clearly $\mathbf{a} < \mathbf{a}'$ holds for all $\mathbf{a}$. Thus, starting with any Turing degree $\mathbf{a}$, we have an ascending sequence of Turing degrees

$$\mathbf{a} < \mathbf{a}' < \mathbf{a}'' < \cdots < \mathbf{a}^{(n)} < \mathbf{a}^{(n+1)} < \cdots$$

In particular, starting with the zero Turing degree $\mathbf{0}$, we have the ascending sequence

$$\mathbf{0} < \mathbf{0}' < \mathbf{0}'' < \cdots < \mathbf{0}^{(n)} < \mathbf{0}^{(n+1)} < \cdots$$

corresponding to the arithmetical hierarchy.

1. Given Turing oracles f and g, prove that the following conditions are pairwise equivalent:

   (a) $f \leq_T g$

   (b) $H^f \leq_m H^g$

   (c) all partial f-recursive functions are partial g-recursive

   (d) all total f-recursive functions are g-recursive.

2. Use finite approximations to construct Turing degrees $\mathbf{a}, \mathbf{b}$ such that $\mathbf{a} > \mathbf{0}$ and $\mathbf{b} > \mathbf{0}$ and $\inf(\mathbf{a}, \mathbf{b}) = \mathbf{0}$.

3. Use finite approximations to construct Turing degrees $\mathbf{a}, \mathbf{b}$ such that $\mathbf{a} < \mathbf{0}'$ and $\mathbf{b} < \mathbf{0}'$ and $\sup(\mathbf{a}, \mathbf{b}) = \mathbf{0}'$.

4. Combine and generalize Exercises 2 and 3 to prove the following:

   > Given two Turing degrees $\mathbf{c}, \mathbf{d}$ such that $\mathbf{c}' \leq \mathbf{d}$, we can find two Turing degrees $\mathbf{a}, \mathbf{b}$ such that $\inf(\mathbf{a}, \mathbf{b}) = \mathbf{c}$ and $\sup(\mathbf{a}, \mathbf{b}) = \mathbf{d}$.

5. Prove the following result.

   > Given an ascending sequence of Turing degrees
   >
   > $$\mathbf{d}_0 < \mathbf{d}_1 < \cdots < \mathbf{d}_n < \mathbf{d}_{n+1} < \cdots$$

we can find a pair of Turing degrees $\mathbf{a}, \mathbf{b}$ such that for all Turing degrees $\mathbf{c}$

$$\exists n \, (\mathbf{c} \leq \mathbf{d}_n) \quad \text{if and only if} \quad \mathbf{c} \leq \mathbf{a} \text{ and } \mathbf{c} \leq \mathbf{b}\,.$$

6. Use the result of Exercise 5 to prove that no ascending sequence of Turing degrees has a least upper bound.

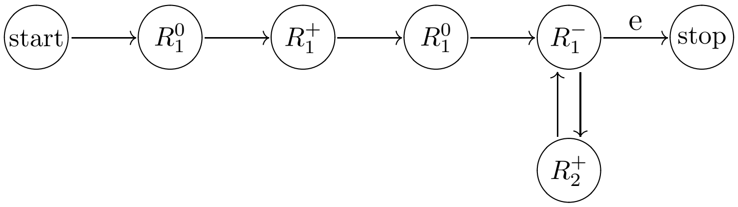
7. For any pair of Turing degrees $\mathbf{a}, \mathbf{b}$ as in Exercise 5, prove that the greatest lower bound $\inf(\mathbf{a}, \mathbf{b})$ does not exist.

Lecture 10: September 17, 2007

17 Oracle computation

We extend the power of register machines by considering *oracle machines*. We already have the increment instruction R_i^+ and the decrement instruction R_i^-; we now add the oracle instruction R_i^0 which replaces n in R_i with $f(n)$ for some fixed total function $f : \mathbb{N} \to \mathbb{N}$, called a *Turing oracle*.

Example 17.1. Given a total function $f(x)$, consider the function $g(x) = f(f(x)+1)$. We claim that g is computable using f as an oracle. This is shown by the oracle program



which uses f as an oracle to compute g.

Definition 17.2. We extend our Gödel numbering system to oracle programs. The definition of $\#(I_m)$ for an oracle instruction I_m is $\#(I_m) = 4 \cdot 3^i \cdot 5^n$ where I_m is an R_i^0 instruction pointing to I_n as the next instruction. The definition of $\#(\mathcal{P})$ is then as before. Thus we now have a Gödel numbering

$$\# : \{\text{oracle programs}\} \xrightarrow{1-1} \mathbb{N}$$

extending our earlier Gödel numbering

$$\# : \{\text{programs}\} \xrightarrow{1-1} \mathbb{N}.$$

Notation 17.3. If $\mathcal{P}$ is an oracle program and f is an oracle, we denote by $\mathcal{P}^f(x_1, \ldots, x_k)$ the run of $\mathcal{P}$ using oracle f starting with $x_1, \ldots, x_k$ in $R_1, \ldots, R_k$ and all other registers empty. The *output* of $\mathcal{P}^f(x_1, \ldots, x_k)$ is the content of R_{k+1} if and when $\mathcal{P}^f(x_1, \ldots, x_k)$ halts. Let $e = \#(\mathcal{P})$. We write

$$\varphi_e^{(k),f}(x_1, \ldots, x_k) \quad \simeq \quad \text{the output of } \mathcal{P}^f(x_1, \ldots, x_k)\,.$$

We denote by W_e^f the domain of $\varphi_e^{(1),f}$.

Definition 17.4. Let f be a fixed oracle. A partial function ψ is *partial f-recursive*, or *partial recursive relative to f*, if there exists e such that

$$\psi(x_1,\dots,x_k) \quad \simeq \quad \varphi_e^{(k),f}(x_1,\dots,x_k) \text{ for all } x_1,\dots,x_k \in \mathbb{N}\,.$$

Example 17.5. Let $f(x) = \chi_H = \begin{cases} 1 & \text{if } \varphi_x^{(1)}(0)\downarrow \\ 0 & \text{if } \varphi_x^{(1)}(0)\uparrow \end{cases}$ Then clearly the halting problem H is solvable relative to f. Further, all r.e. sets are f-recursive.

Proof. Given an r.e. set $W_e = \{x \mid \varphi_e^{(1)}(x)\downarrow\}$, define the function $\theta(x,y) = \varphi_e^{(1)}(x)$. By the parametrization theorem, there is a total recursive function h such that $\theta(x,y) \simeq \varphi_{h(x)}^{(1)}(y)$.

In particular, this is true for $y = 0$; so $\varphi_e^{(1)}(x) \simeq \varphi_{h(x)}^{(1)}(0)$.

Then $\chi_{W_e} = \chi_H \circ h$, which is f-recursive. The oracle program will contain the program for h, then will run the oracle instruction on $h(x)$ to see if it halts. □

Example 17.6. If the oracle f is itself a computable function, then the f-computable functions are exactly the computable functions.

18 Relativization

Theorem 18.1 (Enumeration Theorem). Let f be a fixed oracle. For each $k \geq 0$, the $(k+1)$-place partial function

$$(e, x_1, \dots, x_k) \mapsto \varphi_e^{(k),f}(x_1,\dots,x_k)$$

is partial f-recursive.

Proof. Define the functions $\text{State}^f(e,x_1,\dots,x_k,n)$, $\text{NextState}^f(e,z)$, and $\text{StopTime}^f(e,x_1,\dots,x_k)$ as before. These functions are partial f-recursive. The definition of $\text{NextState}(e,z)$ has an extra clause for oracle instructions:

$\text{NextState}^f(e,z) = z \cdot p_0^{-m+n} \cdot p_i^{-x+f(x)}$
whenever $(z)_0 = m$ and $(e)_m = 4 \cdot 3^i \cdot 5^n$ and $(z)_i = x$.

Then as before we have

$$\varphi_e^{(k),f}(x_1,\dots,x_k) \simeq (\text{State}^f(e,x_1,\dots,x,\text{StopTime}^f(e,x_1,\dots,x_k)))_{k+1}.$$

□

Theorem 18.2 (Parametrization Theorem). Given a 2-place partial f-recursive function $\psi(w,x)$, we can find a 1-place total recursive function $h(w)$ such that $\varphi_{h(w)}^{(1),f}(x) \simeq \psi(w,x)$ for all w,x.

Note: We can find a function h that is recursive, not merely f-recursive.

Proof. Relativize the proof given earlier. □

Definition 18.3 (Relativized Arithmetical Hierarchy). Let f be a fixed oracle. For $k, n \geq 1$, a k-place predicate $P(-)$ is $\Sigma_n^{0,f}$ if $P(-)$ can be written in the form

$$P(-) \equiv \exists y_1 \, \forall y_2 \cdots \begin{matrix} \exists \\ \forall \end{matrix} \, y_n R(-, y_1, y_2, \ldots, y_n)$$

where R is an f-recursive predicate.

(Similarly define $\Pi_n^{0,f}$ and $\Delta_n^{0,f}$.)

Theorem 18.4. A set S is $\Sigma_1^{0,f}$ if and only if $S = W_e^f$ for some e.

Proof. Relativization. □

19 Turing degrees

Definition 19.1. Given two functions $f, g : \mathbb{N} \to \mathbb{N}$, we say f *is Turing reducible to* g, denoted $f \leq_T g$, if there is an integer e such that $f(x) = \varphi_e^{(1),g}(x)$ for all x. (i.e., f is g-recursive.)

The relation $\leq_T$ is reflexive and transitive.

1. $f \leq_T f$
2. $f \leq_T g, g \leq_T h \Rightarrow f \leq_T h$

   Pf: If $\mathcal{P}$ computes f using g as an oracle, and $\mathcal{Q}$ computes g using h as an oracle, then construct a program $\mathcal{R}$ with oracle h as follows:

   Starting with the program $\mathcal{P}$, but replace every oracle instruction R_i^0 with the program $\mathcal{Q}$ (modified so that it takes its input from R_i and outputs back into R_i). Then $\mathcal{R}$ computes f with oracle h.

Definition 19.2. Two functions $f, g : \mathbb{N} \to \mathbb{N}$ are *Turing equivalent* if $f \leq_T g$ and $g \leq_T f$. In this case, we write $f \equiv_T g$. Note that $\equiv_T$ is an equivalence relation.

Definition 19.3 (Degrees of unsolvability). The *Turing degrees* are the set of $\equiv_T$ equivalence classes. If $f : \mathbb{N} \to \mathbb{N}$, then $\deg_T(f) = \{g : \mathbb{N} \to \mathbb{N} \mid f \equiv_T g\}$ is the Turing degree of f.

The Turing degrees are partially ordered by Turing reducibility.

$$f \leq_T g \Leftrightarrow \deg_T(f) \leq_T \deg_T(g).$$

Lectures 11,12: September 19 and 20, 2007

Turing degrees are partially ordered by Turing reducibility. This partial ordering is an object of great interest in the study of unsolvable problems, and many papers have been written investigating its properties.

Basic properties:

1. There is a least Turing degree $\mathbf{0} = \{f : \mathbb{N} \to \mathbb{N} \mid f \text{ is recursive.}\}$.

2. Any two Turing degrees have a least upper bound.

   Proof. Let $\mathbf{a} = \deg_T(f)$ and $\mathbf{b} = \deg_T(g)$. Their least upper bound is $\sup(\mathbf{a}, \mathbf{b}) = \deg_T(f \oplus g)$ ($\mathbf{a}$ join $\mathbf{b}$) where

   $$\begin{cases} f \oplus g(2n) = f(n) \\ f \oplus g(2n+1) = g(n) \end{cases}$$

   Clearly, $f \leq_T f \oplus g$ and $g \leq_T f \oplus g$. Suppose $f \leq_T h$ and $g \leq_T h$; then $f \oplus g \leq_T h$ by using the program for computing f on the even values and the program for computing g on the odd values. □

3. Not every pair of Turing degrees has a greatest lower bound $\inf(\mathbf{a}, \mathbf{b})$ (the *infimum* or *meet* of $\mathbf{a}$ $\mathbf{b}$).

   Proof. The construction of such a pair of Turing degrees is a homework problem. □

4. For each $f : \mathbb{N} \to \mathbb{N}$, there is a set $A \subseteq \mathbb{N}$ such that $\deg_T(A) := \deg_T(\chi_A) = \deg_T(f)$.

   So, without loss of generality, we could just study the Turing degrees of sets.

   Proof. Define $A = \{3^x \cdot 5^{f(x)} \mid x \in \mathbb{N}\} = G_f$, the "graph" of f.

   $$\chi_A \leq_T f, \text{ since } y \in A \equiv \underbrace{\left(\underbrace{y = 3^{(y)_1} \cdot 5^{(y)_2}}_{\text{recursive}} \wedge \underbrace{f((y)_1) = (y)_2}_{f-\text{recursive}} \right)}_{f-\text{recursive}}.$$

   Note also that $f \leq_T \chi_A$, since $f(x) = \mu y\, (3^x \cdot 5^y \in A)$. □

Problem: Given a Turing degree $\mathbf{a} = \deg_T(f)$, find a Turing degree strictly greater than $\mathbf{a}$. (In other words, given a function f, find a function g that is not f-recursive.)

Definition 19.4. The *Halting problem relative to* f is $H^f = \{x \mid \varphi_x^{(1),f}(0) \downarrow\}$.

The *Turing jump operator* maps $\begin{array}{rcl} \deg_T(f) & \mapsto & \deg_T(H^f) \\ \mathbf{a} & \mapsto & \mathbf{a}' \end{array}$

Properties:

1. H^f is $\Sigma_1^{0,f}$ complete.
2. $f <_T H^f$ ($f \leq_T H^f$ and $H^f \not\leq_T f$)
3. $f \leq_T g \Rightarrow H^f \leq_T H^g$
4. In particular, $\mathbf{a}' \geq_T \mathbf{0}'$ for all $\mathbf{a}$

We present two theorems, without proof for now, that describe further the structure of Turing degrees and the Turing jump operator.

Theorem 19.5 (Friedberg's Jump Theorem). For all Turing degrees $\mathbf{c} > \mathbf{0}'$ ($\mathbf{0}' = \deg(H)$, the Turing degree of the halting problem), there exists a Turing degree $\mathbf{a}$ such that $\mathbf{a}' = \mathbf{c}$.

In other words, the range of the Turing jump operator is $\{\mathbf{c} \mid \mathbf{c} \geq \mathbf{0}'\}$.

Proof. See pages 31-33 of Spring 2004 lecture notes. □

20 Finite approximation

Definition 20.1. A *string* is a finite sequence of natural numbers.

Notation 20.2. We use Greek letters such as $\sigma, \tau, \ldots$ to denote strings. The length of a string σ will be denoted $|\sigma|$. The elements of σ will be denoted $\sigma(i)$ for $i < |\sigma|$. So, if $|\sigma| = m$, then

$$\sigma = \langle \sigma(0), \sigma(1), \ldots, \sigma(m-1) \rangle .$$

We write $\sigma \subseteq \tau$ if and only if $|\sigma| \leq |\tau|$ and $\sigma(i) = \tau(i)$ for all $i < |\sigma|$. The *concatenation* of σ and τ is

$$\sigma^\frown\tau = \langle \sigma(0), \sigma(1), \ldots, \sigma(m-1), \tau(0), \tau(1), \ldots, \tau(n-1) \rangle$$

where $|\sigma| = m$ and $|\tau| = n$. Note that $\sigma^\frown\tau$ is a string of length $m + n$.

Since strings are finite objects, they can be Gödel numbered. For concreteness we choose the Gödel numbering

$$\#(\sigma) = \prod_{i<|\sigma|} p_i^{\sigma(i)+1}$$

noting that

$$\# : \{\text{strings}\} \xrightarrow{1-1} \mathbb{N} .$$

For example, if $\sigma = \langle 3, 8, 11 \rangle$, then $|\sigma| = 3$, $\sigma(0) = 3$, $\sigma(1) = 8$, $\sigma(2) = 11$, and $p_0 = 2$, $p_1 = 3$, $p_2 = 5$, so $\#(\sigma) = 2^4 3^9 5^{12}$.

Note that the set of Gödel numbers of strings is recursive (use bounded quantification, etc.). Moreover, $|\sigma|$ is recursive as a function of $\#(\sigma)$, and $\#(\sigma^\frown\tau)$ is recursive as a function of $\#(\sigma)$ and $\#(\tau)$. Also, $\sigma(i)$ is recursive as a function of $\#(\sigma)$ and i, $\sigma \subseteq \tau$ is recursive as a predicate on $\#(\sigma)$ and $\#(\tau)$, etc.

We now introduce the important concept of finite approximations.

If f is an oracle, let

$$f \upharpoonright n = \langle f(0), f(1), \ldots, f(n-1) \rangle \ .$$

We call this string $f \upharpoonright n$ a *finite approximation* of f. Note that f is the union of its finite approximations. We write $\sigma \subset f$ to mean that the string σ is a finite approximation of the oracle f, i.e., $\sigma = f \upharpoonright n$ for some n.

In general, an oracle contains an infinite amount of information: $f(0), f(1), f(2), f(3), f(4), \ldots$. However, in any particular halting computation

$$\varphi_e^{(1),f}(x) \quad \simeq \quad y$$

only finitely much information from the oracle is used, since the computation halts in only a finite number of steps and consults the oracle only a finite number of times. We use the following notation to describe oracle computations that use only a finite approximation to f.

Notation 20.3. $\varphi_{e,s}^{(k),\sigma}(x_1, \ldots, x_k) \simeq y$ means:

> for any or all oracles f approximated by σ, $\varphi_e^{(k),f}(x_1, \ldots, x_k) \simeq y$ in less than or equal to s steps of computation using only oracle information from σ.

We write $\varphi_e^{(k),\sigma}(x_1, \ldots, x_k) \simeq y$ as an abbreviation for $\varphi_{e,|\sigma|}^{(k),\sigma}(x_1, \ldots, x_k) \simeq y$.

Lemma 20.4. The following are pairwise equivalent:

1. $\varphi_e^{(k),f}(x_1, \ldots, x_k) \simeq y$.

2. $\varphi_{e,s}^{(k),f\upharpoonright n}(x_1, \ldots, x_k) \simeq y$ for some n and s.

3. $\varphi_{e,s}^{(k),f\upharpoonright n}(x_1, \ldots, x_k) \simeq y$ for all sufficiently large n and s.

4. $\varphi_e^{(k),f\upharpoonright n}(x_1, \ldots, x_k) \simeq y$ for some n.

5. $\varphi_e^{(k),f\upharpoonright n}(x_1, \ldots, x_k) \simeq y$ for all sufficiently large n.

6. $\varphi_{e,s}^{(k),\sigma}(x_1, \ldots, x_k) \simeq y$ for some string $\sigma \subset f$ and some s.

7. $\varphi_{e,s}^{(k),\sigma}(x_1, \ldots, x_k) \simeq y$ for all sufficiently long strings $\sigma \subset f$ and all sufficiently large s.

8. $\varphi_e^{(k),\sigma}(x_1, \ldots, x_k) \simeq y$ for some string $\sigma \subset f$.

9. $\varphi_e^{(k),\sigma}(x_1, \ldots, x_k) \simeq y$ for all sufficiently long strings $\sigma \subset f$.

Proof. Obvious. □

Note also that the above-mentioned predicates on strings have a monotonicity property:

Lemma 20.5. As usual let us write $-$ instead of $x_1, \ldots, x_k$.

1. If $\sigma \subseteq \tau$ and $s \leq t$, then $\varphi_{e,s}^{(k),\sigma}(-) \simeq y$ implies $\varphi_{e,t}^{(k),\tau}(-) \simeq y$.
2. If $\sigma \subseteq \tau$ and $s \leq t$, then $\varphi_{e,s}^{(k),\sigma}(-)\downarrow$ implies $\varphi_{e,t}^{(k),\tau}(-)\downarrow$.
3. If $\sigma \subseteq \tau$ then $\varphi_{e}^{(k),\sigma}(-) \simeq y$ implies $\varphi_{e}^{(k),\tau}(-) \simeq y$.
4. If $\sigma \subseteq \tau$ then $\varphi_{e}^{(k),\sigma}(-)\downarrow$ implies $\varphi_{e}^{(k),\tau}(-)\downarrow$.

Proof. Obvious. □

Some of the usefulness of finite approximations lies in the fact that certain predicates associated with them are recursive. Namely:

Lemma 20.6. Let us write σ instead of $\#(\sigma)$, the Gödel number of σ. As usual we write $-$ instead of $x_1, \ldots, x_k$.

1. The $(k+4)$-place predicate $\{\langle e, s, \sigma, -, y\rangle \mid \varphi_{e,s}^{(k),\sigma}(-) \simeq y\}$ is recursive.
2. The $(k+3)$-place predicate $\{\langle e, \sigma, -, y\rangle \mid \varphi_{e}^{(k),\sigma}(-) \simeq y\}$ is recursive.
3. The $(k+3)$-place predicate $\{\langle e, s, \sigma, -\rangle \mid \varphi_{e,s}^{(k),\sigma}(-)\downarrow\}$ is recursive.
4. The $(k+2)$-place predicate $\{\langle e, \sigma, -\rangle \mid \varphi_{e}^{(k),\sigma}(-)\downarrow\}$ is recursive.

Proof. Let $\mathrm{NextState}^{\sigma}(e, z)$ be the finite approximation version of the NextState function. Note that$\mathrm{NextState}^{\sigma}(e, z)$ is recursive as a function of the three variables $\#(\sigma), e, z$. The only difference is that its definition now has an extra clause for oracle instructions:

$$\mathrm{NextState}^{\sigma}(e, z) = z \cdot p_0^{-m+n} \cdot p_i^{-x+\sigma(x)}$$
whenever $(z)_0 = m$ and $(e)_m = 4 \cdot 3^i \cdot 5^n$ and $(z)_i = x$ and $x < |\sigma|$.

By recursion on n, it follows that $\mathrm{State}^{\sigma}(e, x_1, \ldots, x_k, n)$ is recursive as a function of $\#(\sigma), e, x_1, \ldots, x_k, n$. Hence, the predicates

$$\begin{aligned}\varphi_{e,s}^{(k),\sigma}(x_1, \ldots, x_k) \simeq y \quad \equiv \quad & (\exists n \leq s)\,[\,(\mathrm{State}^{\sigma}(e, x_1, \ldots, x_k, n))_0 = 0 \\ & \wedge (\mathrm{State}^{\sigma}(e, x_1, \ldots, x_k, n))_{k+1} = y\,]\end{aligned}$$

and

$$\varphi_{e,s}^{(k),\sigma}(x_1, \ldots, x_k)\downarrow \quad \equiv \quad (\exists n \leq s)\,[\,(\mathrm{State}^{\sigma}(e, x_1, \ldots, x_k, n))_0 = 0\,]$$

etc., are recursive. □

The method of finite approximation is a valuable tool in studying the structure of the Turing degrees. For example:

Theorem 20.7 (Kleene/Post). There exist incomparable Turing degrees below $\mathbf{0}'$. That is, there are $\mathbf{a}, \mathbf{b} \leq \mathbf{0}'$ such that $\mathbf{a} \not\leq \mathbf{b}$ and $\mathbf{a} \not\leq \mathbf{b}$.

Proof. By finite approximation.

We will build sets $A, B \subseteq \mathbb{N}$ so that $\mathbf{a} = \deg_T A$ and $\mathbf{b} = \deg_T B$, by defining longer and longer strings $\sigma_n, \tau_n \in 2^{<\mathbb{N}}$ with $\sigma_n \subseteq \sigma_{n+1}$ and $\tau_n \subseteq \tau_{n+1}$. Then $f = \chi_A = \displaystyle\bigcup_{n=1}^{\infty} \sigma_n$ and $g = \chi_B = \displaystyle\bigcup_{n=1}^{\infty} \tau_n$.

The condition $\mathbf{a} \not\leq \mathbf{b}$ requires that for all e, $f \neq \varphi_e^{(1),g}$; the condition $\mathbf{b} \not\leq \mathbf{a}$ requires that for all e, $g \neq \varphi_e^{(1),f}$. We will take care of one of these infinitely many requirements at each step of our construction.

Start with $\sigma_0, \tau_0 = \langle\rangle$ (the empty string); this is stage $n = 0$.

Stage $n = 2e + 1$:

Case 1: If it exists, find the least (by Gödel number) $\sigma \in 2^{<\mathbb{N}}$ extending σ_{2e} such that $\varphi_e^{(1),\sigma}(|\tau_{2e}|) \simeq 1$. Then set $\sigma_{2e+1} = \sigma$ and $\tau_{2e+1} = \tau_{2e}{}^\frown\langle 0\rangle$.

Case 2: Otherwise, set $\sigma_{2e+1} = \sigma 2e$ and $\tau_{2e+1} = \tau_{2e}{}^\frown\langle 1\rangle$.

Stage $n = 2e + 2$:

(same as odd-numbered stages, but with σ and τ switched)

Case 1: If it exists, find the least (by Gödel number) $\tau \in 2^{<\mathbb{N}}$ extending τ_{2e+1} such that $\varphi_e^{(1),\tau}(|\sigma_{2e+1}|) \simeq 1$. Set $\tau_{2e+2} = \tau$ and $\sigma_{2e+2} = \sigma_{2e+1}{}^\frown\langle 0\rangle$.

Case 2: Otherwise, set $\tau_{2e+2} = \tau 2e + 1$ and $\sigma_{2e+2} = \sigma_{2e+1}{}^\frown\langle 1\rangle$.

Then define $f = \chi_A = \displaystyle\bigcup_{n=1}^{\infty} \sigma_n$ and $g = \chi_B = \displaystyle\bigcup_{n=1}^{\infty} \tau_n$.

Claim: $f \not\leq_T g$

Take any $e \in \mathbb{N}$. We will show that it is not the case that $f \simeq \varphi_e^{(1),g}$. Consider step $2e + 2$. Let $x = |\sigma_{2e+1}|$.

If $f(x) = 0$, then step $2e + 2$ was case 1. Then $\varphi_e^{(1),g}(x) \simeq \varphi_e^{(1),\tau}(x) \simeq 1$, so $f \not\simeq \varphi_e^{(1),g}$.

If $f(x) = 1$, then step $2e+2$ was case 2. That means there was no extension τ of τ_{2e+1} so that $\varphi_e^{(1),\tau}(x) \simeq 1$. But g is an extension of τ_{2e+1}, so we know that $f \not\simeq \varphi_e^{(1),g}$.

Claim: $g \not\leq_T f$

(mirror the above argument)

Claim: $\deg_T(f) \leq \mathbf{0}'$ and $\deg_T(g) \leq \mathbf{0}'$

The whole construction is recursive relative to the Halting problem ($\mathbf{0}'$ recursive).

Everything is recursive except the division into cases 1 and 2. The search for extensions σ and τ might never halt. Use $\mathbf{0}'$ to determine if this search will halt. If it does, use stage 1; if it does not, use stage 2.

□

Lecture 13: September 21, 2007

21 Comments on Homework #4

Homework #4, Problem 1

Problem 1 is an exercise in understanding Turing reducibility, many-one reducibility, and the Turing jump operator. Basically, the problem is to prove that $f \leq_T g$ if and only if $H^f \leq_m H^g$.

Recall that

$$\begin{aligned} H^f &= \text{ the Halting Problem relative to the oracle } f \\ &= \{x \mid \varphi_x^{(1),f}(0)\downarrow\} \end{aligned}$$

is the "relativization to f" of $H = \{x \mid \varphi_x^{(1)}(0)\downarrow\}$ = the Halting Problem.

We have already implicitly proved that the Halting Problem is unsolvable, but let us now make this explicit:

Theorem 21.1. *H is Σ^0_1 complete, hence not recursive.*

Proof. We used the Parametrization Theorem to prove that H is Σ^0_1-complete. It follows that H is not Δ^0_1, hence not recursive. □

We now relativize this to H^f to obtain:

Theorem 21.2. *H^f is not f-recursive. I.e., $H^f \not\leq_T f$. Note also that $f \leq_T H^f$*

Proof. The point is that H^f is $\Sigma^{0,f}_1$ complete, hence not $\Delta^{0,f}_1$, hence not $\leq_T f$. This is just the relativization to f of the previous theorem.

It remains to show that $f \leq_T H^f$. By definition, for any set $A \subseteq \mathbb{N}$, $f \leq_T A$ means $f \leq_T \chi_A$, also $\deg_T(A) = \deg_T(\chi_A)$, etc.

For any function f, define the "graph" of f to be the set $G_f = \{3^x 5^y \mid f(x) = y\}$. Then $f \equiv_T G_f$, because $f(x) = \mu y\, 3^x 5^y \in G_f$, and $z \in G_f \equiv (\exists x < z)\,(\exists y < z)\,(z = 3^x 5^y \wedge y = f(x))$. It follows that G_f is $\Delta^{0,f}_1$, so it is $\Sigma^{0,f}_1$. Hence $G_f \leq_m H^f$, hence $G_f \leq_T H^f$, hence $f \leq_T H^f$. □

Summary: For any Turing oracle f, we have $H^f >_T f$.

In the above proof, we used the fact that many-one reducibility is a special case of Turing reducibility. Formally,

Lemma 21.3. *$A \leq_m B$ implies $A \leq_T B$.*

Proof. Suppose $A \leq_m B$ via a function h. Then $\chi_A(x) = \chi_B(h(x))$. h is recursive, so χ_A is χ_B-recursive. □

Definition 21.4. The *Turing jump operator* is the operator which takes f to H^f.

Thus the Turing jump operator is simply the relativization of the Halting Problem. If $\mathbf{a} = \deg_T(f)$, then we write $\mathbf{a}' = \deg_T(H^f)$. Note that $\mathbf{a} < \mathbf{a}'$ for all Turing degrees $\mathbf{a}$. In other words, given any unsolvable problem, the jump operator gives us a problem which is "more unsolvable," i.e., its degree of unsolvability is greater.

In particular, starting with the Turing degree $\mathbf{0} = \deg_T(0)$ where 0 denotes any recursive function, we may iterate the jump operator to obtain an ascending sequence of Turing degrees $\mathbf{0} < \mathbf{0}' < \mathbf{0}'' < \mathbf{0}''' < \cdots < \mathbf{0}^{(n)} < \mathbf{0}^{(n+1)} < \cdots$ which represent higher and higher degrees of unsolvability. It can be shown that this sequence of Turing degrees is closely related to the arithmetical hierarchy. See the discussion of Post's Theorem, below.

To solve Problem 1, use the following facts:

1. H^f is complete $\Sigma_1^{0,f}$.

2. A set A is $\Sigma_1^{0,f}$ if and only if A is the range of a partial f-recursive function.

3. A 1-place partial function $\psi(x)$ is partial f-recursive if and only if G_ψ, the graph of ψ, is $\Sigma_1^{0,f}$. Here of course $G_\psi = \{3^x 5^y \mid \psi(x) \simeq y\}$.

All of these are straightforward relativizations of known facts about Σ_1^0 sets and partial recursive functions.

22 Homework #5, due October 1, 2007

Exercises 22.1.

1. Recall that a simple r.e. set is neither recursive nor many-one complete. Use Post's Theorem plus relativization to generalize this to higher levels of the arithmetical hierarchy.

   Conclude that for each $n \geq 1$ there exist Σ_n^0 sets which are neither many-one complete (within the class of Σ_n^0 sets) nor Δ_n^0.

2. (a) Given a Σ_1^0 predicate $P(x, y)$ such that $\forall x\, \exists y\, P(x, y)$ holds, prove that there exists a recursive function $f(x)$ such that $\forall x\, P(x, f(x))$ holds.

   (b) Use Post's Theorem plus relativization to generalize the previous result to higher levels of the arithmetical hierarchy.

   Conclude that for all $n \geq 1$, given a Σ_n^0 predicate $P(x, y)$ such that $\forall x\, \exists y\, P(x, y)$ holds, there exists a Δ_n^0 function $f(x)$ such that $\forall x\, P(x, f(x))$ holds.

3. Prove the following:

   (a) Every infinite recursively enumerable set includes an infinite recursive set.

(b) Every infinite recursive set includes a recursively enumerable set which is not recursive.

(c) Every infinite recursive set is the union of two disjoint infinite recursive sets.

(d) Every infinite recursively enumerable set is the union of two disjoint infinite recursively enumerable sets.

(e) (Extra Credit) Every recursively enumerable set which is nonrecursive is the union of two disjoint recursively enumerable sets which are nonrecursive.

4. Given a nonrecursive recursively enumerable set A, prove that we can find a simple set B such that $A \equiv_T B$.

   Hint: Use a deficiency set.

5. Prove the followng theorem:

   > Given a Turing degree $\mathbf{d} \geq \mathbf{0}'$, we can find a Turing degree $\mathbf{a}$ such that $\mathbf{a}' = \mathbf{d}$.

   Thus, the range of the Turing jump operator consists precisely of the Turing degrees which are $\geq \mathbf{0}'$.

   Hint: Use the technique of finite approximation.

Lecture 14: September 24, 2007

23 Comments on Homework #4, continued

Finite approximation: additional explanation

A basic method in the study of Turing degrees is finite approximation. Let σ be a string. The technique of finite approximation says: $\varphi_e^{(1),g}(x) \simeq y$ iff

$$(\exists s)\,(\exists \sigma \text{ a finite approximation to } g) \qquad \underbrace{[\varphi_{e,s}^{(1),\sigma}(x) \simeq y]}_{\text{this predicate is recursive!}} \qquad .$$

The finite approximation method consists of defining functions f, g by $f = \bigcup_{n=0}^{\infty} \sigma_n$ and $g = \bigcup_{n=0}^{\infty} \tau_n$ where $\sigma_0 \subseteq \sigma_1 \subseteq \sigma_2 \subseteq \cdots \subseteq \sigma_n \subseteq \cdots$ and $\tau_0 \subseteq \tau_1 \subseteq \tau_2 \subseteq \ldots \subseteq \tau_n \subseteq \ldots$. This method was originally introduced by Kleene and Post.

At each step n, we choose extensions σ_{n+1} and τ_{n+1} to accomplish some requirement which we want to hold for f, g.

As a first application of the method, we prove the following theorem.

Theorem 23.1. There exist Turing degrees $\mathbf{a}, \mathbf{b}$ such that $\mathbf{a} \nleq \mathbf{b}$ and $\mathbf{b} \nleq \mathbf{a}$. (I.e., they are incomparable.)

In other words, there exist 1-place total functions f, g such that $f \not\leq_T g$ and $g \not\leq_T f$.

Note that the requirement $f \not\leq_T g$ can be broken down into a countable family of requirements. For each index e, we require: $f(x) \not\simeq \varphi_e^{(1),g}(x)$ for some x. This requirement can be satisfied in either of two ways: $\varphi_e^{(1),g}(x) \uparrow$, or $\varphi_e^{(1),g}(x) \downarrow \neq f(x)$.

Proof. We shall construct f and g by finite approximation. This means that, by induction on n, we shall construct infinite increasing sequences of strings

$$\sigma_0 \subseteq \sigma_1 \subseteq \sigma_2 \subseteq \sigma_3 \subseteq \ldots \subseteq \sigma_n \subseteq \ldots$$

and

$$\tau_0 \subseteq \tau_1 \subseteq \tau_2 \subseteq \tau_3 \subseteq \ldots \subseteq \tau_n \subseteq \ldots$$

and after that we shall define $f = \bigcup_{n=0}^{\infty} \sigma_n$ and $g = \bigcup_{n=0}^{\infty} \tau_n$.

The construction:

Stage 0: Let $\sigma_0 = \tau_0 = \langle\rangle =$ the empty string $=$ the unique string of length 0.

Stage $2e+1$: We may assume inductively that σ_{2e} and τ_{2e} are already known. Let $x = |\sigma_{2e}|$.

Case 1: $\exists s\, \exists \tau \supseteq \tau_{2e}$ such that $\varphi_{e,s}^{((1),\tau}(x) \downarrow$. In this case, choose such s, τ. (For example, we could choose the least such pair (s, τ) according to some fixed Gödel numbering of pairs.) Then define $y = \varphi_{e,s}^{(1),\tau}(x)$ and $\sigma_{2e+1} = \sigma_{2e}{}^\frown\langle y+1\rangle$ and $\tau_{2e+1} = \tau$.

Case 2: Not case 1. In this case, let $\sigma_{2e+1} = \sigma_{2e}$ and $\tau_{2e+1} = \tau_{2e}$.

Stage $2e+2$: We proceed as in Stage $2e+1$ except that the roles of f and g are reversed. Here are the details. We may assume inductively that σ_{2e+1} and τ_{2e+1} are already known. Let $x = |\tau_{2e+1}|$.

Case 1: $\exists s\, \exists \sigma \supseteq \sigma_{2e+1}$ such that $\varphi_{e,s}^{((1),\sigma}(x) \downarrow$. In this case, choose such s, σ and define $y = \varphi_{e,s}^{(1),\sigma}(x)$ and $\tau_{2e+2} = \tau_{2e+1}{}^\frown\langle y+1\rangle$ and $\sigma_{2e+2} = \sigma$.

Case 2: Not case 1. In this case, let $\sigma_{2e+2} = \sigma_{2e+1}$ and $\tau_{2e+2} = \tau_{2e+1}$.

This completes the construction.

As already mentioned, we now define $f = \bigcup_{n=0}^{\infty} \sigma_n$ and $g = \bigcup_{n=0}^{\infty} \tau_n$.

We claim that $f \not\leq_T g$ and $g \not\leq_T f$. To prove the claim, we argue by contradiction.

Suppose for instance that $f \leq_T g$. Then, there exists e such that $f(x) = \varphi_e^{(1),g}(x)$ for all x. For this particular e, consider what happened at stage $2e+1$ of the construction. Let x be as in stage $2e+1$, i.e., $x = |\sigma_{2e}|$. For this particular x, let n and s be sufficiently large so that $\varphi_{e,s}^{(1),g\restriction n}(x) \downarrow$ and $n \geq |\tau_{2e}|$. Letting $\tau = g \restriction n$, we see that $\tau_{2e} \subseteq \tau$ and $\varphi_{e,s}^{(1),\tau}(x) \downarrow$. Thus we see that Case 1 holds. Therefore, let s and τ be as chosen in Case 1, and let $y = \varphi_{e,s}^{(1),\tau}(x)$. (Note that this s and τ may be different from the previous s and τ). Then, by construction, we have $f(x) = \sigma_{2e+1}(x) = y+1$ and $\tau_{2e+1} = \tau$ and $\varphi_e^{(1),g}(x) = \varphi_{e,s}^{(1),\tau_{2e+1}}(x) = y$, a contradiction. Thus we have proved that $f \not\leq_T g$.

The proof that $g \not\leq_T f$ is similar, looking at stage $2e+2$ instead of stage $2e+1$. □

Solution of Homework #4, Problem 2

The problem is to find $\mathbf{a}, \mathbf{b}$ such that $\mathbf{a}, \mathbf{b} > \mathbf{0}$ and $\inf(\mathbf{a}, \mathbf{b}) = \mathbf{0}$.

So, we want to find f, g such that $f \not\leq_T g$ and $g \not\leq_T f$ (this requirement is as before) and in addition, if $h \leq_T f$ and $h \leq_T g$ then h is recursive.

Thus, for each pair of indices i, j we have a new requirement:

> if $\varphi_i^{(1),f}(x)$ is total and $\varphi_j^{(1),g}(x)$ is total and they are the same total function, then that function is recursive.

Here are some possible ways to bring this about at stage n:

1. $\exists x \neg \exists \sigma \supseteq \sigma_n \, \exists s \, \varphi_{i,s}^{(1),\sigma}(x) \downarrow$.

   [In this case, $\varphi_i^{(1),f}$ is not total.]

2. $\exists x \neg \exists \tau \supseteq \tau_n \, \exists s \, \varphi_{j,s}^{(1),\tau}(x) \downarrow$.

   [In this case, $\varphi_j^{(1),g}(x)$ is not total.]

3. $\exists x \, \exists \sigma \supseteq \sigma_n \, \exists \tau \supseteq \tau_n \, \exists s \, \varphi_{i,s}^{(1),\sigma}(x) \downarrow \neq \varphi_{j,s}^{(1),\tau}(x) \downarrow$.

   [This insures that they are not the same total function.]

4. $\varphi_i^{(1),f}(x)$ is total and recursive.

Now for the actual construction:
Stage $n+1$ where $n = 4 \cdot 3^i 5^j$:
We are given σ_n, τ_n.
Case 1: $\exists x \, \exists \sigma \supseteq \sigma_n \, \exists \tau \supseteq \tau_n \, \exists s \, \varphi_{i,s}^{(1),\sigma}(x) \downarrow \neq \varphi_{j,s}^{(1),\tau}(x) \downarrow$
In this case choose σ, τ as above and let $\sigma_{n+1} = \sigma$ and $\tau_{n+1} = \tau$.
Case 2: Not case 1.
In this case let $\sigma_{n+1} = \sigma_n$ and $\tau_{n+1} = \tau_n$.
Finally let $f = \bigcup_{n=0}^{\infty} \sigma_n$ and $g = \bigcup_{n=0}^{\infty} \tau_n$.

We need to prove that our requirement is satisfied. Suppose $h(x) = \varphi_i^{(1),f}(x) = \varphi_j^{(1),g}(x)$ are the same total function. We need to prove that this function is recursive.

To see this, let $n = 4 \cdot 3^i 5^j$ and consider what happened at stage $n+1$ of the construction. If Case 1 happened, then we have an x such that $\varphi_i^{(1),\sigma_{n+1}}(x) \downarrow \neq \varphi_j^{(1),\tau_{n+1}}(x) \downarrow$, hence $\varphi_i^{(1),f}(x)$ and $\varphi_j^{(1),g}(x)$ could not be the same total function. So, Case 2 must have happened at stage $n+1$. In this case we claim that $h(x)$ is computable. Namely, given x, to compute $h(x)$, search for $\sigma \supseteq \sigma_n$ and s such that $\varphi_{i,s}^{(1),\sigma}(x) \downarrow$. Then $h(x) = \varphi_{i,s}^{(1),\sigma}(x)$, since there is exactly one possible value that we can get, no matter which extension σ we choose. If there were more than one possible value, then Case 1 would have happened at this stage.

This completes our sketch of the solution of Homework #4, Problem 2.

Lecture 15: September 26, 2007

Hint for Homework #4, Problem 3

We want $\sup(\mathbf{a}, \mathbf{b}) = \mathbf{0}'$. Construct f, g by finite approximation. We need

1. $f \not\leq_T g$ and $g \not\leq_T f$.
2. $f \oplus g \leq_T H$
3. $H \leq_T f \oplus g$

We have done (1).

Essentially, we have done (2). In our finite approximation constructions, $f = \bigcup_{n=1}^{\infty} \sigma_n$ where $\sigma_0 \subseteq \sigma_1 \subseteq \ldots \subseteq \sigma_n \subseteq_{n+1} \subseteq \ldots$

Stage n+1: Given σ_n, construct σ_{n+1}. We divide our constructions into two cases:

Case 1: $\exists \sigma \supseteq \sigma_n \, \exists x \, \exists s \, (\varphi_{e,s}^{(1),\sigma}(x) \downarrow)$

Case 2: Not case 1

Note that the division into cases is not computable. However, it is Σ^0_1. Since the Halting Problem is Σ^0_1 complete, the division into cases is computable relative to the Halting Problem ($\mathbf{0}'$). Thus, the construction of f is computable from $\mathbf{0}'$; that is $f \leq_T H$.

Hint for Homework #4, Problems 4 and 5

Problem 4 combines and generalizes problems 2 and 3. We need $\inf(\mathbf{a}, \mathbf{b}) = \mathbf{0}$ and $\sup(\mathbf{a}, \mathbf{b}) = \mathbf{0}'$. Then relativize to get $\inf(\mathbf{a}, \mathbf{b}) = \mathbf{c}$ and $\sup(\mathbf{a}, \mathbf{b}) = \mathbf{d}$. Instead of finite approximation, use coinfinite approximation.

Definition 23.2. A *coinfinite condition* is a 1-place partial function, p, such that $\operatorname{dom}(p)$ is recursive and coinfinite.

For problems 4 and 5, construct sequences of coinfinite conditions $p_0 \subseteq p_1 \subseteq p_2 \subseteq \cdots \subseteq p_n \subseteq p_{n+1} \subseteq \cdots$ and let $f = \bigcup_{n=0}^{\infty} p_n$. Similarly, construct $q_0 \subseteq q_1 \subseteq q_2 \subseteq \ldots \subseteq q_n \subseteq \ldots$ and let $g = \bigcup_{n=0}^{\infty} q_n$. Here $p \subseteq q$ means $\operatorname{dom}(p) \subseteq \operatorname{dom}(q)$ and $p(x) = q(x)$ for all $x \in \operatorname{dom}(p)$. It is also useful to define $p \subseteq_{\text{fin}} q$ to mean $p \subseteq q$ and $\operatorname{dom}(q) \setminus \operatorname{dom}(q)$ is finite.

For problem 4 use coinfinite conditions which are $\leq_T C$ where $\mathbf{c} = \deg_T(C)$.

For problem 5 use coinfinite conditions which are $\leq_T D_n$ for some n, where $\mathbf{d}_n = \deg_T(D_n)$.

24 Parametrization and uniformity

Recall the concept of indices. If $e = \#(\mathcal{P})$, we say that e is an *index* of the partial recursive function $\varphi_e^{(1)}(x)$ which is computed by $\mathcal{P}$. We also say that e is an *index* of the r.e. set $W_e = \operatorname{dom}(\varphi_e^{(1)}) = \{x \mid \varphi_e^{(1)}(x) \downarrow\}$.

The true meaning of the Parametrization Theorem is embodied in the following vague but useful "uniformity principle":

> Many operations on partial recursive functions and r.e. sets can be described as computable operations on the indices of these functions and sets.

Example 24.1. Consider the union of two r.e. sets, $W_x \cup W_y$. We know that this is r.e., hence there exists z such that $W_z = W_x \cup W_y$. The Uniformity Principle tells us something more. Namely, given x and y we can compute an index $z = f(x, y)$ of $W_x \cup W_y$. Here $f(x, y)$ is a total computable function.

Proof. We have

$$\begin{aligned} u \in W_x \cup W_y \quad &\equiv \quad \exists s\,[\,(\mathrm{State}(x,u,s))_0 = 0 \vee (\mathrm{State}(y,u,s))_0 = 0\,] \\ &\equiv \quad \psi(u,x,y)\downarrow \end{aligned}$$

where $\psi(u,x,y) \simeq \mu s\,[\,(\mathrm{State}(x,u,s))_0 = 0 \vee (\mathrm{State}(y,u,s))_0 = 0\,]$. So far we have shown that $W_x \cup W_y$ is Σ^0_1, i.e., it is an r.e. set. We want to show more, that we can find its index computably from x and y.

By the Parametrization Theorem, find a recursive function $g(w)$ such that $\varphi^{(1)}_{g(w)}(u) \simeq \psi(u, (w)_1, (w)_2)$ for all u, w. Consider the recursive function $f(x, y) = g(3^x 5^y)$. Then $\varphi^{(1)}_{f(x,y)}(u) \simeq \varphi^{(1)}_{g(3^x5^y)}(u) \simeq \psi(u, x, y)$, hence $\varphi^{(1)}_{f(x,y)}(u)\downarrow \equiv u \in W_x \cup W_y$. Thus $W_{f(x,y)} = W_x \cup W_y$ for all x, y as desired. □

Example 24.2. Similarly, we can show that various other operations on indices are recursive. For instance, we obtain total recursive functions f, g, h, k, l with the following properties.

1. $W_{f(x,y)} = W_x \cup W_y$
2. $W_{g(x,y)} = W_x \cap W_y$
3. $W_{h(x,y)} = W_x \cup \{y\}$
4. $\varphi^{(1)}_{k(x,y)} = \varphi^{(1)}_x \circ \varphi^{(1)}_y$
5. $W_{l(x,y)} = \left(\varphi^{(1)}_x\right)^{-1}(W_y) = \{u \mid \varphi^{(1)}_x(u)\downarrow \in W_y\}$

We now apply the Uniformity Principle to solve some of the problems in Homework #3.

Solution of Homework #3, Problem 2(b)

Show that $A \leq_m B$, A creative, B r.e. imply B creative. It suffices to show that if $P \leq_m Q$ and P is productive, then Q is productive.

Proof. Let $h(x)$ be a productive function for P. This means that if $W_x \subseteq P$ then $h(x) \notin W_x$ and $h(x) \in P$. Assume $P \leq_m Q$ via the recursive function f, i.e., $x \in P$ if and only if $f(x) \in Q$. Suppose now that $W_x \subseteq Q$. Then $f^{-1}(W_x)$ is an r.e. set; further, by the uniformity principle (Parametrization Theorem), there is a recursive function $g(x)$ such that $W_{g(x)} = f^{-1}(W_x)$ for all x. We have $W_{g(x)} \subseteq P$ since $W_x \subseteq Q$, so we apply our productive function for P to get $h(g(x)) \in P, \notin W_{g(x)}$. Finally, apply the reduction function f to get $k(x) = f(h(g(x))) \in Q, \notin W_x$. Thus $k = f \circ h \circ g$ is a productive function for Q. □

Solution of Homework #3, Problem 3(b)

Show that a creative set is not simple. It suffices to show that a productive set P is not immune. We need to show that P has an infinite r.e. set.

Let $h(x)$ be a productive function for P. Start with x_0, an index of $\emptyset$, the empty set. Then $W_{x_0} = \emptyset$, so trivially $W_{x_0} \subseteq P$. Hence $h(x_0) \notin W_{x_0}$ (obviously) and $h(x_0) \in P$. Let $W_{x_1} = \{h(x_0)\} \subseteq P$. Then $h(x_1) \notin W_{x_1}$ and $h(x_1) \in P$. Continuing in this fashion, we generate a sequence of distinct integers $h(x_0), h(x_1), h(x_2), \ldots$ which hopefully form an infinite r.e. subset of P.

To make this work, we need to show the construction of $x_0, x_1, x_2, \ldots$ can be done in a uniform or recursive manner, via the Parametrization Theorem. The Uniformity Principle gives us a recursive function $g(x, y)$ such that $W_{g(x,y)} = W_x \cup \{y\}$ for all x, y. We want $W_{x_{n+1}} = W_{x_n} \cup \{h(x_n)\} = W_{g(x_n, h(x_n))}$ for all n, so define the infinite sequence $x_0, x_1, x_2, \ldots$ by letting $x_{n+1} = g(x_n, h(x_n))$ for all n. Since $h(x)$ and $g(x, y)$ are recursive functions, the sequence $x_0, x_1, \ldots, x_n, \ldots$ is recursive. It follows that the set $\{h(x_0), h(x_1), h(x_2), \ldots\}$ is an infinite r.e. subset of P.

Lecture 16: September 28, 2007

25 Review for the upcoming Midterm Exam

Some topics to study are:

1. r.e. sets (various characterizations)
2. creative sets, simple sets, deficiency sets
3. Post's Theorem
4. Parametrization Theorem, uniformity
5. the diamond: $$\begin{array}{ccc} & \mathbf{0}' & \\ / & & \backslash \\ \mathbf{a} & & \mathbf{b} \\ \backslash & & / \\ & \mathbf{0} & \end{array}$$
6. coinfinite approximation

Post's Theorem

As usual abbreviate a k-place predicate $P(x_1, \ldots, x_k)$ as $P(-)$.

Theorem 25.1 (Post's Theorem). $P(-)$ is Σ^0_n if and only if $P(-)$ is $\Sigma_1^{0,\mathbf{0}^{(n-1)}}$.

$n = 1$: trivial.
$n = 2$: $P(-)$ is $\Sigma^0_2 \Leftrightarrow P(-)$ is Σ^0_1 relative to $\mathbf{0}' =$ the Halting Problem.
An interesting consequence is: A is Δ^0_2 if and only if $A \leq_T$ the Halting Problem.

Proof. We omit the proof; it is by finite approximation to $\mathbf{0}^{(n-1)}$. □

The point of Post's Theorem is that many properties of Σ^0_1 sets easily relativize to become properties of Σ^0_n sets. Of course, we understand Σ^0_1 sets very well, because they are the same thing as recursively enumerable sets. Post's Theorem says that our detailed understanding of Σ^0_1 sets applies also to higher levels of the arithmetical hierarchy.

Deficiency sets

Consider a $1-1$ total recursive function f. If f is monotone increasing ($x < y \Rightarrow f(x) < f(y)$), then the range of f is recursive (not just Σ^0_1). Namely,

$$\begin{aligned} z \in \text{range of } f &\equiv \exists x f(x) = z \\ &\equiv \exists x \leq z f(x) = z \end{aligned}$$

In general, the range of a recursive $1-1$ function need not be recursive. In fact, any nonrecursive r.e. set is the range of a 1-1 total recursive function.

We define the *deficiency set of* f to be $D_f = \{x \mid \exists y\,(x < y \wedge f(y) < f(x)\}$. The set D_f measures the failure of f to be monotone increasing. Note that D_f is Σ^0_1 by definition. It is easy to show that if $\text{rng}(f)$ is recursive then D_f is recursive. Moreover, if $\text{rng}(f)$ is nonrecursive, then D_f is norecursive, and in fact D_f is simple.

Let us show that if $\text{rng}(f)$ is nonrecursive then D_f is simple. We have to show three things:

1. D_f is Σ^0_1 (obvious).
2. D_f is coinfinite.
3. D_f is not disjoint from any infinite r.e. set.

If D_f is cofinite, then for all sufficiently large x we have $x \in D_f$, which means $\exists y > x\, f(y) < f(x)$. Choose x_0 to be such a sufficiently large x. Then we can find $x_1 > x_0$ with $f(x_1) < f(x_0)$. Then we can find $x_2 > x_1$ with $f(x_2) < f(x_1)$. Then we can find $x_3 > x_2$ with $f(x_3) < f(x_2)$. .... This gives an infinite descending sequence of natural numbers: $f(x_0) > f(x_1) > f(x_2) > \cdots$. No such sequence exists, so D_f is coinfinite.

Now suppose $B \cap D_f = \emptyset$, where B is infinite Σ^0_1. We then argue that $\text{rng}(f)$ is recursive. Namely, to decide whether $z \in \text{rng}(f)$, search for $x \in B$ such that

$f(x) > z$. But $x \in B$ implies $x \notin D_f$, i.e., $f(y) > f(x)$ for all $y > x$. So, $z \in \text{rng}(f) \equiv \exists y \leq x \, z = f(y)$.

Here is a more formal version of the proof. Suppose B is Σ^0_1 and disjoint from D_f. Since B is Σ^0_1, we have $x \in B \equiv \exists y \, R(x, y)$, where R is recursive. Consider the total recursive function $g(z) = (\mu w \, (R((w)_1, (w)_2) \wedge f((w)_1) > z))_1$. Then $g(z) = x$ where $x \in B$ and $f(x) > z$. Then, as we found above, $z \in \text{rng}(f) \equiv [z = f(y)$ for some $y \leq g(z)]$.

A problem in Homework #5 is: Given a nonrecursive r.e. set A, find a simple set B such that $A \equiv_T B$.

Here is a hint. By our characterization of r.e. sets, let f be a 1-1 recursive function whose range is A. Then we may let $B = D_f$. The proof that $A \equiv_T D_f$ is somewhat similar to the above proof that D_f is simple.

Coinfinite conditions

A *coinfinite condition* is a 1-place partial function p such that dom(p) is recursive and coinfinite. The technique of coinfinite approximation is a variant of the technique of finite approximation by strings. For a coinfinite condition p define $\varphi^{(1),p}_{e,s}(x) \simeq y$ just as we did for strings, namely this means that the program with Gödel number e and input x using oracle p halts in $\leq s$ steps with output y consulting the oracle only for values in dom(p). Some useful notations for coinfinite conditions are:

1. $p \subseteq q$ means: $\text{dom}(p) \subseteq \text{dom}(q)$ and $p(x) = q(x)$ for all $x \in \text{dom}(p)$.

2. $p \subseteq_{\text{fin}} q$ means: $p \subseteq q$ and $\text{dom}(q) \setminus \text{dom}(p)$ is finite.

As an example of how to use coinfinite approximation, consider the proof of the following result concerning Turing degrees:

> For all $\mathbf{c}$ there exist $\mathbf{a}, \mathbf{b} > \mathbf{c}$ such that $\inf(\mathbf{a}, \mathbf{b}) = \mathbf{c}$.

This is the relativization to $\mathbf{c}$ of the unrelativized result that there exist Turing degrees $\mathbf{a}, \mathbf{b} > \mathbf{0}$ such that $\inf(\mathbf{a}, \mathbf{b}) = \mathbf{0}$. The unrelativized result was proved with finite approximations. To prove the relativization, use coinfinite approximations which are $\leq_T h$ for some fixed function h such that $\deg_T(h) = \mathbf{c}$. We will have $\mathbf{a} = \deg_T(f)$, $\mathbf{b} = \deg_T(g)$, $f =$ union of $p_0 \subseteq_{\text{fin}} p_1 \subseteq_{\text{fin}} p_2 \subseteq_{\text{fin}} \cdots$, $g =$ union of $q_0 \subseteq_{\text{fin}} q_1 \subseteq_{\text{fin}} q_2 \subseteq_{\text{fin}} \cdots$. To guarantee $\mathbf{a}, \mathbf{b} \geq \mathbf{c}$, start with p_0 and q_0 defined as follows: $p_0(2n) = h(n)$ and $p_0(2n+1) \uparrow$ for all n, and $q_0 = p_0$. Note that p_0 and q_0 are coinfinite conditions which include an infinte amount of information, namely h. We use finite extensions of p_0 and q_0 to accomplish $\mathbf{a} \not\leq \mathbf{b}$ and $\mathbf{b} \not\leq \mathbf{a}$ and $\inf(\mathbf{a}, \mathbf{b}) = \mathbf{c}$. This is similar to how we earlier used finite approximations to get incomparable $\mathbf{a}, \mathbf{b}$ with $\inf(\mathbf{a}, \mathbf{b}) = \mathbf{0}$.

Coinfinite conditions are useful for Homework #4, Problems 4 and 5. For Problem 4 use coinfinite conditions which are $\leq_T C$ where $\deg_T(C) = \mathbf{c}$. For Problem 5 use coinfinite conditions which are $\leq_T D_n$ for some n, where $\deg_T(D_n) = \mathbf{d}_n$.

26 Midterm Exam, October 3, 2007

Exercises 26.1.

1. Explicitly exhibit a set which is Π^0_5 and not Σ^0_5.

2. We have seen that, given a 1-place partial recursive function ψ which is one-to-one, the inverse function ψ^{-1} is again partial recursive. The Uniformity Principle tells us that, given an index of ψ, we should expect to be able to compute an index of ψ^{-1}.

   (a) Give a rigorous statement of this result concerning indices.

   (b) Give a full proof of this result, using the Parametrization Theorem.

3. Let A and B be subsets of $\mathbb{N}$. If A and B are simple, prove that $A \cap B$ is simple.

4. Let A, B, C be recursively enumerable subsets of $\mathbb{N}$ such that $A = B \cup C$ and $B \cap C = \emptyset$. Let $\mathbf{a}, \mathbf{b}, \mathbf{c}$ be the respective Turing degrees of A, B, C. Prove that $\mathbf{a} = \sup(\mathbf{b}, \mathbf{c})$.

5. Consider the sets $R = \{e \mid W_e \text{ is recursive}\}$, $C = \{e \mid W_e \text{ is creative}\}$, and $S = \{e \mid W_e \text{ is simple}\}$. What can you say or guess in the way of classifying R, C and S in the arithmetical hierarchy? Prove as much as you can.

6. (a) Let f_i, $i = 0, 1, 2, \ldots$ be a countable sequence of nonrecursive total 1-place functions. Use the method of finite approximation to construct a nonrecursive total 1-place function g such that $f_i \not\leq_T g$ for all i.

   (b) Deduce that for any Turing degree $\mathbf{a} > \mathbf{0}$ we can find a Turing degree $\mathbf{b} > \mathbf{0}$ such that $\inf(\mathbf{a}, \mathbf{b}) = \mathbf{0}$.

27 Homework #6, due October 8, 2007

Exercise 27.1. For each natural number n define

$$C_\varphi(n) \quad = \quad \mu e \left(\varphi^{(1)}_e(0) \simeq n \right) .$$

Intuitively, $C_\varphi(n)$ is the smallest "description" of n in terms of our standard enumeration of the 1-place partial recursive functions, $\varphi^{(1)}_e$, $e = 0, 1, 2, \ldots$. Note that C_φ is a total 1-place function, but it is not recursive.

Consider the set

$$S \quad = \quad \{n \mid C_\varphi(n) < \log\log\log n\} .$$

Intuitively, S is the set of all n such that n has a relatively small description. For example, the number

$n =$ (10 to the 10 to the 10 to the 10 to the $1,000,000,000$ power)

belongs to S because, although it is very large, it is also very easy to describe.

Prove that S is a simple set. This means:

1. S is recursively enumerable.

2. The complement of S is infinite.

3. The complement of S includes no infinite recursively enumerable set.

Lecture 17: October 1, 2007

28 Solutions of some homework problems

Solution of Homework #5, Problem 3

A simple fact is that, for any infinite set A, π_A (the principal function of A) is $\equiv_T A$. (Recall that $A = \{\pi_A(0) < \pi_A(1) < \pi_A(2) < \cdots < \pi_A(n) < \cdots\}$.)

In particular, if A is nonrecursive then π_A is nonrecursive.

For Problem 3(c), let S be infinite recursive, so π_S is recursive. Define $S_1 = \{\pi_S(n) \mid n \text{ odd}\}$, $S_2 = \{\pi_S(n) \mid n \text{ even}\}$. Then S_1 and S_2 are infinite recursive, and $S = S_1 \cup S_2$ and $S_1 \cap S_2 = \emptyset$.

Problem 3(d) is similar. Let A be r.e. infinite. Then $A = \text{rng}(f)$ for some 1-1 total recursive f. Let $A_1 = \{f(n) \mid n \text{ odd}\}$ and $A_2 = \{f(n) \mid n \text{ even}\}$. Then $A = A_1 \cup A_2$ and A_1, A_2 are r.e. infinite and $A_1 \cap A_2 = \emptyset$.

For Problem 3(a), let A and f be as above. Define a recursive increasing function g by $g(0) = f(0)$, $g(i+1) = f(\mu n\, f(n) > g(i))$. Then g is recursive and by definition $g(i+1) > g(i)$ for all i. Letting $B = \text{rng}(g)$ we have $g = \pi_B$, $B \subseteq A$, B is recursive and infinite.

Here is an alternative solution, from Robin Tucker-Drob, inspired by deficiency sets. Let $C = \{f(n) \mid f(n) \geq n\}$. We have $m \in C \equiv (\exists n \leq m)\,(m = f(n))$ so C is recursive. We can argue that C is infinite.

For Problem 3(b), let S be an infinite recursive set. Let $B = \{\pi_S(n) \mid n \in K\}$. Clearly B is an infinite r.e. subset of S. Also B is nonrecursive, because for all n we have $n \in K \Leftrightarrow \pi_S(n) \in B$, so $K \leq_m B$.

Problem 3(e) is harder; the result is a published paper.

Solution of Homework #5, Problem 5

The problem was to prove the following theorem of Friedberg:

Given a Turing degree $\mathbf{d} \geq \mathbf{0}'$, to find a Turing degree $\mathbf{a}$ such that $\mathbf{a}' = \mathbf{d}$.

In other words, given $g \geq_T H$ where H is the Halting Problem, to find f such that $H^f \equiv_T g$.

We construct f by finite approximations, $f = \bigcup_{n=0}^{\infty} \sigma_n$, $\sigma_0 \subseteq \sigma_1 \subseteq \cdots \subseteq \sigma_n \subseteq \cdots$. Our strategy is, at even-numbered stages control H^f, and at odd-numbered stages code in g.

Stage 0: Let $\sigma_0 = \langle\rangle$, the empty string.

Stage $2e+1$: Let $\sigma_{2e+1} = \sigma_{2e}{}^\frown\langle g(e)\rangle$. The purpose here is to code in some information about g.

Stage $2e+2$:

Case 1: $(\exists\sigma \supseteq \sigma_{2e+1})\,(\varphi_e^{(1),\sigma}(0)\downarrow)$. In this case choose the least such σ (according to the Gödel numbering of strings) and let $\sigma_{2e+2} = \sigma$. Thus we have forced $e \in H^f$, recalling that $H^f = \{e \mid \varphi_e^{(1),f}(0)\downarrow\}$.

Case 2: Not case 1. In this case let $\sigma_{2e+2} = \sigma_{2e+1}$. The purpose of this case distinction is to control the jump of f. The construction insures that $e \in H^f$ if and only if Case 1 held at Stage $2e+2$.

We claim that the *entire construction* (i.e., the 1-place total function c where $c(n) = \#(\sigma_n)$ for all n) is $\leq_T$ each of the oracles g, $H \oplus f$, H^f.

To prove the claim, let $J = \{\langle e,\tau\rangle \mid (\exists\sigma \supseteq \tau)\,(\varphi_e^{(1),\sigma}(0)\downarrow)\}$. This J is exactly the oracle that we need in order to distinguish between Case 1 and Case 2 at Stage $2e+2$. Namely, we are in Case 1 if and only if $\langle e, \sigma_{2e+1}\rangle \in J$. Once we know which case we are in, we can compute σ_{2e+2} recursively given σ_{2e+1}. Now observe that J is Σ_1^0. Since H is Σ_1^0 complete, it follows that $J \leq_m H$, hence $J \leq_T H$. Since $H \leq_T g$, it follows that the entire construction is $\leq_T g$. Also, the entire construction is $\leq_T H \oplus f$, because at Stage $2e+1$ we have $\sigma_{2e+1} = \sigma_{2e}{}^\frown\langle f(|\sigma_{2e}|)\rangle$. Since the entire construction is $\leq_T H \oplus f$, it is also $\leq_T H^f$, because obviously $H \oplus f \leq_T H^f$. This completes the proof of the claim.

Note also that $H^f \leq_T$ the entire construction. This is clear, because $e \in H^f$ if and only if Case 1 held at Stage $2e+2$, i.e., if and only if $\varphi_e^{(1),\sigma_{2e+2}}(0)\downarrow$.

Finally, note that $g \leq_T$ the entire construction, because $g(e) = \sigma_{2e+1}(|\sigma_{2e}|)$.

Putting everything together, we have $g \equiv_T H \oplus f \equiv_T H^f \equiv_T$ the entire construction. Letting $\mathbf{a} = \deg_T(f)$, it follows that $\mathbf{d} = \sup(\mathbf{0}', \mathbf{a}) = \mathbf{a}'$. This completes the proof.

Solution of Homework #5, Problem 2

Problem 2(a): Given a Σ_1^0 predicate $P(x,y)$ such that $\forall x\,\exists y\,P(x,y)$, to find a recursive *selector*, i.e., a recursive function $f(x)$ such that $\forall x\,P(x, f(x))$.

A wrong construction is $f(x) = \mu y\,P(x,y)$. The wrong thing about this is that we are applying the μ operator to a nonrecursive predicate.

A correct construction is as follows. Let $P(x,y) \equiv \exists z\,R(x,y,z)$ where R is a recusive predicate. Let $g(x) =$ least $3^y \cdot 5^z$ such that $R(x,y,z)$ holds. Then, let $f(x) = (g(x))_1$.

Problem 2(b): Using Post's Theorem, this generalizes as follows. Given a Σ_n^0 predicate $P(x,y)$ such that $\forall x\,\exists y\,P(x,y)$, we can find a Δ_n^0 selector.

Note also that we can drop the hypothesis $\forall x\,\exists y\,P(x,y)$. In this case the same construction gives a partial recursive selector, i.e., a partial recursive function $\psi(x)$ such that for all x, if $\exists y\,P(x,y)$ then $\psi(x)$ is defined and is such a y.

In the generalization to Σ^0_n predicates, we get a selector whose graph is Σ^0_n.

29 Homework #7, due October 15, 2007

Exercises 29.1.

1. Exhibit an oracle program $\mathcal{P}$ such that

$$\varphi_e^{(1),f}(x) \quad \simeq \quad \mu y\,(y > x \wedge f(y) = 0)$$

for all $f \in \mathbb{N}^{\mathbb{N}}$ and all $x \in \mathbb{N}$, where $e = \#(\mathcal{P})$.

2. (a) Give an explicit example of a Δ^0_4 set which is neither Σ^0_3 nor Π^0_3.

   (b) Give an example of a set which cannot be classified in the arithmetical hierarchy.

3. Let A, B, C be recursively enumerable sets with $A = B \cup C$ and $B \cap C = \emptyset$. If $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are the respective Turing degrees of A, B, C prove that $\mathbf{a} = \sup(\mathbf{b}, \mathbf{c})$.

   Note: The hardest part is to prove that $B \leq_T A$ and $C \leq_T A$. Your proof should use the assumption that A, B, C are r.e. sets. Without this assumption, the result would not be correct.

4. Construct an infinite descending sequence of Turing degrees

$$\mathbf{a}_0 > \mathbf{a}_1 > \cdots > \mathbf{a}_n > \mathbf{a}_{n+1} > \cdots$$

or prove that no such sequence exists.

5. Let $C(\sigma)$ denote the Kolmogorov complexity of a $0,1$-valued string σ.

   We have seen in class that

$$C(\sigma^\frown\tau) \ \leq \ 2C(\sigma) + 2C(\tau) + O(1)$$

for all $0,1$-valued strings σ and τ. Improve this inequality to

$$C(\sigma^\frown\tau) \ \leq \ C(\sigma) + 2\log_2 C(\sigma) + C(\tau) + O(1)$$

where $\log_2 x$ denotes the base 2 logarithm of x.

Can you make further improvements?

Lecture 18: October 8, 2007

30 Solutions of some homework and midterm problems

Solution of Homework #5, Problem 1

The problem is to prove that for each $n \geq 1$ there exist Σ_n^0 sets which are neither Δ_n^0 nor Σ_n^0-complete.

Proof. For $n = 1$, let A be a simple set. We have seen in a previous homework that A is not recursive and not Σ_1^0-complete.

For $n = 2$, it does not work to take A' = the jump of A, where A is a simple set. The set A' is indeed Σ_2^0, but it could be Σ_2^0-complete. For instance, suppose A is a simple set which is $\equiv_T K$. Such A's exist by Problem 4. Then $A' \equiv_m K'$ is Σ_2^0-complete.

For a correct solution, use the following definition.

Definition 30.1. A set A is *n-simple* if A is Σ_n^0, A^c is infinite, and A^c does not include an infinite Σ_n^0 set.

Post's Theorem says that Σ_n^0 is the same as $\Sigma_1^{0,\mathbf{0}^{(n-1)}}$. By relativizing the usual facts about simple sets to $\mathbf{0}^{(n-1)}$, we see that there exists an n-simple set and any such set is neither Σ_n^0-complete nor Δ_n^0. □

Solution of Homework #5, Problem 2

Given a Σ_1^0 predicate $P(x, y)$ such that $\forall x \, \exists y \, P(x, y)$, let $P(x, y) \equiv \exists z \, R(x, y, z)$ with R recursive and define

$$f(x) \simeq (\mu w \, R(x, (w)_1, (w)_2))_1 \, .$$

Then clearly $f(x)$ is recursive and $\forall x \, P(x, f(x))$.

Now, imitate the above argument with Σ_n^0 instead of Σ_1^0. This will give a direct solution without using Post's Theorem. Assume that $P(x, y)$ is Σ_n^0 and $\forall x \, \exists y \, P(x, y)$. Then $P(x, y) \equiv \exists z \, R(x, y, z)$ where R is Π_{n-1}^0. Define $f(x)$ as above. We need to show that f is Δ_n^0. We have $f(x) = y \equiv \exists w \, [\, (R(x, (w)_1, (w)_2) \wedge \neg \, \exists v < w \, R(x, (v)_1, (v)_2)) \wedge y = (w)_1 \,]$ so the graph of f is Σ_n^0. Similarly $f(x) = y \equiv \forall w \, [\, R(x, (w)_1, (w)_2) \wedge \neg \, \exists v < w \, R(x, (v)_1, (v)_2) \,] \Rightarrow y = (w)_1 \,]$ so the graph of f is Π_n^0. Thus the graph of f is Δ_n^0, i.e., f is Δ_n^0.

Midterm, Problem 6

Let f_i, $i = 0, 1, 2, \ldots$ be a sequence of nonrecursive functions. Then we can find g nonrecursive such that $f_i \not\leq_T g$ for $i = 0, 1, 2, \ldots$. This is done by finite approximation: $g = \bigcup_{n=0}^{\infty} \tau_n$ where $\tau_0 \subseteq \tau_1 \subseteq \cdots$.

Stage 0. Let $\tau_0 = \langle\rangle$.

Stage $2e+1$. Let $x = |\tau_{2e}|$. If $\varphi_e^{(1)}(x)\downarrow$ let $\tau_{2e+1} = \tau_{2e}{}^\frown\langle\varphi_e^{(1)}(x)+1\rangle$. Otherwise do nothing, i.e., let $\tau_{2e+1} = \tau_{2e}$.

Stage $2e+2$ where $e = 3^i 5^j$.

Case 1: $(\exists\tau \supseteq \tau_{2e+1})\,\exists x\,(\varphi_j^{(1),\tau}(x)\downarrow\neq f_i(x))$. In this case choose such a τ and let $\tau_{2e+2} = \tau$.

Case 2: Not case 1. In this case do nothing, i.e., let $\tau_{2e+2} = \tau_{2e+1}$.

Stage $2e+1$ insures that g is not recursive with index e. Stage $2e+2$ insures that $f_i \neq \varphi_j^{(1),g}$ because otherwise f_i would be recursive. Namely, for all x we would have $f_i(x) = \varphi_j^{(1),\tau}(x)$ for the least $\tau \supseteq \tau_{2e+1}$ such that $\varphi_j^{(1),\tau}(x)\downarrow$.

Midterm, Problem 3

We are to prove that if A and B are simple then $A \cap B$ is simple. The proof is based on the following lemma.

Lemma 30.2. If A is simple, then every infinite r.e. set W_e has an infinite intersection with A.

Proof. Supose W_e infinite and $W_e \cap A = F$ finite. Then $W_e \setminus F$ is also infinite and r.e. However, $(W_e \setminus F) \cap A \neq \emptyset$ since A is simple. This contradiction proves the lemma. □

To see that $A \cap B$ is simple, let W_e be an infinite r.e. set. Since A is simple, we see by the previous lemma that $W_e \cap A$ is infinite. This is again an infinite r.e. set. Since B is simple, it follows that $W_e \cap A \cap B \neq \emptyset$, Q.E.D.

Midterm, Problem 5

Define

1. $R = \{e \mid W_e \text{ recursive}\}$
2. $C = \{e \mid W_e \text{ creative}\}$
3. $S = \{e \mid W_e \text{ simple}\}$

The problem is to classify R, C, S in the arithmetical hierarchy.

To classify R we have

$$\begin{aligned} e \in R \quad &\equiv \quad \exists i\ [W_i \text{ is the complement of } W_e] \\ &\equiv \quad \exists i\,[W_i \cap W_e = \emptyset \wedge W_i \cup W_e = \mathbb{N}] \\ &\equiv \quad \underbrace{\exists i\ \underbrace{\forall x\ \underbrace{\left[\underbrace{x \notin W_i \cap W_e}_{\Pi_1^0} \wedge \underbrace{x \in W_i \cup W_e}_{\Sigma_1^0}\right]}_{\Delta_2^0}}_{\Pi_2^0}}_{\Sigma_3^0}\,. \end{aligned}$$

Thus R is Σ^0_3 by this Tarski/Kuratowski computation.

Similarly for C we have

$$\begin{array}{rcl} e \in C & \equiv & \exists \, \text{total recursive } h(x) \forall x \; [W_x \cap W_e = \emptyset \Rightarrow h(x) \notin W_x \cup W_e] \\ & \equiv & \exists i \left(\left[\forall x \, \varphi_i^{(1)}(x) \downarrow \right] \wedge \left[\forall x \, W_x \cap W_e \neq \emptyset \vee \forall x \, \forall s \, \varphi_{i,s}^{(1)}(x) \notin W_i \cup W_e \right] \right) \\ & \equiv & \Sigma^0_3 \end{array}$$

A similar computation shows S is Π^0_3.

There is a useful heuristic principle. Namely, if an index set is classified in the arithmetical hierarchy by means of a Tarski/Kuratowski computation as above, then the set “ought to be” many-one complete within that class.

To complete the solution of Problem 5, one must prove that, indeed, R and C are Σ^0_3-complete and S is Π^0_3-complete. This can be done, but the proofs are rather difficult.

Solution of Homework #6

We define $C_\varphi(n) = \mu e \, (\varphi_e^{(1)}(0) \simeq n)$ and $S = \{n \mid C_\varphi(n) < \log \log \log n\}$. The problem is to show that S is a simple set.

1. S is r.e.

   We have $n \in S \;\equiv\; \exists e \, (\underbrace{\underbrace{e < \log^3 n}_{\text{recursive}} \wedge \underbrace{\varphi_e^{(1)}(0) \simeq n}_{\Sigma^0_1}}_{\Sigma^0_1})$.

2. S^c is infinite.

   We know there are infinitely many e such that $\varphi_e^{(1)}(0) \uparrow$. Given k, choose x to be $>$ at least k many such e's. It follows that x is $>$ at least k many n's such that $C_\varphi(n) \geq x$. For these n's we have $n < x \leq C_\varphi(n)$, hence $n \notin S$. Thus S^c includes at least k elements.

3. $S^c \supseteq$ no infinite r.e. set. In other words, “any infinite r.e. set contains elements with short descriptions.”

   By the Parametrization Theorem, let $f(e, x)$ be a recursive function such that
   $$\varphi_{f(e,x)}^{(1)}(0) \simeq \varphi_e^{(1)}(x)$$
   for all e, x. Define a recursive binary relation $\ll$ by
   $$m \ll n \quad \equiv \quad (\forall e \leq m) \, (\forall x \leq m) \, [\, f(e, x) < \log^3(n) \,] \, .$$
   Note that $\forall m \, \exists n \, (m \ll n)$. Now let B be any infinite r.e. set. Let g be a total recursive function such that $B = \text{rng}(g)$. Define a recursive function h by $h(x) = g(\mu y \, (x \ll g(y)))$ for all x. Then for all x we have $x \ll h(x) \in B$. Let e be an index of h. Then for all x we have $h(x) \simeq \varphi_e^{(1)}(x) \simeq \varphi_{f(e,x)}^{(1)}(0)$. In particular, for all $x \geq e$ we have $C_\varphi(h(x)) \leq f(e, x) < \log^3 h(x)$, hence $h(x) \in B \cap S$, Q.E.D.

Lecture 19: October 10, 2007

Advance comments on Homework #7

We comment on each of the problems individually.

1. This is a routine exercise concerning oracle programs. We use the following notation:

   Notation 30.3. $\mathbb{N}^\mathbb{N}$ is the set of all *oracles*, i.e., total 1-place number-theoretic functions.
   $$\mathbb{N}^\mathbb{N} = \{f : \mathbb{N} \to \mathbb{N}\} .$$
   The space $\mathbb{N}^\mathbb{N}$ is called the *Baire space*.

2. This is a routine exercise concerning the arithmetical hierarchy.

3. This is a repeat of one of the midterm problems.

4. This is another exercise on Turing degrees.

5. This exercise concerns Kolmogorov complexity. The idea of Kolmogorov complexity is, if τ is a finite sequence of 0's and 1's, then $C(\tau)$ = the "complexity" of τ, i.e., the "length of the shortest description" of τ.

   For example,
   $$\tau = \langle \overbrace{0, 1, 0, 1, 0, 1, \ldots, 0, 1}^{10^{10^{10^{10}}}} \rangle$$
   is a very long string of 0's and 1's but it has a very short description. The precise definition of $C(\tau)$ is below.

31 Kolmogorov complexity

The purpose of this section is to introduce Kolmogorov complexity. First, some preliminaries.

Notation 31.1. Recall that a *string* is a finite sequence of natural numbers, $\sigma = \langle n_1, n_2, \ldots, n_l \rangle$. Here l = length of $\sigma = |\sigma|$. We write

$$\begin{array}{rcl} \mathbb{N}^{<\mathbb{N}} & = & \text{the set of all strings} \\ & = & \bigcup_{l=0}^{\infty} \mathbb{N}^l . \end{array}$$

A *bitstring* (= 0,1-valued string) is a string of 0's and 1's. For example, $\sigma = \langle 0, 1, 1, 0, 0, 0, 1, 0 \rangle$ is a bitstring of length 8. We write

$$\begin{array}{rclcl} 2^{<\mathbb{N}} & = & \{0,1\}^{<\mathbb{N}} & = & \text{the set of all bitstrings} \\ & = & \bigcup_{l=0}^{\infty} \{0,1\}^l . & & \end{array}$$

We often identify a string σ with its Gödel number $\#(\sigma)$.

Definition 31.2. A *machine* is a partial recursive function from bitstrings to bitstrings:

$$M :\subseteq 2^{<\mathbb{N}} \to 2^{<\mathbb{N}} .$$

A recursive enumeration M_e, $e = 0, 1, 2, \ldots$ of all machines may be obtained from our standard recursive enumeration of all partial recursive functions. Namely we write

$$M_e(\sigma) \simeq \tau \quad \equiv \quad \varphi_e^{(1)}(\#(\sigma)) \simeq \#(\tau)$$

where $\sigma, \tau \in 2^{<\mathbb{N}}$.

Definition 31.3. A *universal machine* is a machine, call it U, with the following property:

$$(\forall \text{ machines } M)\,(\exists \text{ bitstring } \rho)\,(\forall \text{ bitstrings } \sigma)\,[\,U(\rho^\frown\sigma) \simeq M(\sigma)\,] .$$

We may think of the bitstring ρ as a "code" for the machine M in terms of the universal machine U.

Theorem 31.4. Universal machines exist.

Proof. Define

$$U(\langle\underbrace{0,\ldots,0}_{e},1\rangle^\frown\sigma) \quad \simeq \quad M_e(\sigma) .$$

It is easy to check that U is a universal machine. □

Now we define Kolmogorov complexity.

Definition 31.5 (Kolmogorov). Fix a universal machine U. The *complexity* of a bitstring τ is defined as

$$C(\tau) \quad = \quad \min\{|\sigma| \mid U(\sigma) \simeq \tau\} .$$

The idea of this definition is that $C(\tau)$ is supposed to measure the "amount of information" inherent in the bitstring τ. We shall now show that this measure of complexity is, in a sense, independent of the choice of universal machine.

Notation 31.6. $f(x) \leq g(x) + O(1)$ means:

$$\exists c\, \forall x\, (f(x) \leq g(x) + c) .$$

Theorem 31.7. "Up to an additive constant, $C(\tau)$ is well-defined."

The precise statement reads as follows. If $\widehat{U}$ is another universal machine, and if we define $\widehat{C}(\tau) = \min\{|\sigma| \mid \widehat{U}(\sigma) \simeq \tau\}$, then

$$|C(\tau) - \widehat{C}(\tau)| \leq O(1) .$$

Proof. Since $\widehat{U}$ is a machine, let ρ be a bitstring such that $U(\rho^\frown\sigma) \simeq \widehat{U}(\sigma)$ for all σ. Given τ, let $\sigma, \widehat{\sigma}$ be such that $U(\sigma) \simeq \tau$, $\widehat{U}(\widehat{\sigma}) \simeq \tau$, $|\sigma| = C(\tau)$, $|\widehat{\sigma}| = \widehat{C}(\tau)$. We have $\widehat{U}(\widehat{\sigma}) \simeq U(\rho^\frown\widehat{\sigma}) \simeq \tau$, hence $C(\tau) \leq |\rho^\frown\widehat{\sigma}| = |\rho| + |\widehat{\sigma}| = |\rho| + \widehat{C}(\tau)$. Thus $C(\tau) \leq \widehat{C}(\tau) + O(1)$ because ρ is independent of τ. Similarly we can show that $\widehat{C}(\tau) \leq C(\tau) + O(1)$. This completes the proof. □

Notation 31.8. We define the Kolmogorov complexity of an integer as

$$C(n) = C(\langle \underbrace{0, \ldots, 0}_{n} \rangle)\,.$$

Some easy facts are:

1. $C(|\tau|) \leq C(\tau) + O(1)$.
2. $C(\tau) \leq |\tau| + O(1)$.
3. $C(\tau_1{}^\frown\tau_2) \leq 2C(\tau_1) + 2C(\tau_2) + O(1)$.

   Question: Can we improve this to

   $$C(\tau_1{}^\frown\tau_2) \leq C(\tau_1) + C(\tau_2) + O(1) \text{ ?}$$

   This would be more intuitive.

Lecture 20: October 12, 2007

We have defined the *complexity* of a bitstring τ as

$$C(\tau) = \min\{|\sigma| \mid U(\sigma) \simeq \tau\} = \text{“the amount of information in } \tau\text{”}$$

where U is a fixed universal machine. We now prove some simple facts about this notion of complexity.

Proposition 31.9. The following hold for all strings τ. Recall that $f(x) \leq g(x) + O(1)$ means $\exists c\, \forall x\, (f(x) \leq g(x) + c)$.

1. $C(|\tau|) \leq C(\tau) + O(1)$ (where $C(n) = C(\langle \underbrace{0, 0, \ldots, 0}_{n} \rangle)$).

   Proof. Define $M(\sigma) \simeq \langle \underbrace{0, 0, \ldots, 0}_{|U(\sigma)|} \rangle$. Clearly M is a machine. Let ρ be a “code” for M in terms of U. This means that $M(\sigma) \simeq U(\rho^\frown\sigma)$ for all σ. Now, given τ, let σ be such that $U(\sigma) \simeq \tau$ and $|\sigma| = C(\tau)$. Then, for this σ, $U(\rho^\frown\sigma) \simeq M(\sigma) \simeq \langle \underbrace{0, 0, \ldots, 0}_{|U(\sigma)|} \rangle = \langle \underbrace{0, 0, \ldots, 0}_{|\tau|} \rangle$.

   Hence $C(|\tau|) \leq |\rho^\frown\sigma| = |\rho| + |\sigma| = C(\tau) + |\rho| = C(\tau) + O(1)$ since ρ is independent of τ. □

2. $C(\tau) \leq |\tau| + O(1)$.

   Proof. Consider the “identity machine,” $I(\sigma) = \sigma$. Let ρ be a “code” for I, i.e., $U(\rho^\frown\tau) \simeq I(\tau) = \tau$ for all τ. Then $C(\tau) \leq |\rho^\frown\tau| = |\rho| + |\tau| = |\tau| + O(1)$. □

3. $C(\tau_1{}^\frown\tau_2) \leq 2C(\tau_1) + 2C(\tau_2) + O(1)$.

   Proof. Temporarily define a "pairing function" for bitstrings, denoted $\sigma{*}\tau$. If $\sigma = \langle i_1, \ldots, i_m\rangle$ and $\tau = \langle j_1, \ldots, j_n\rangle$, let

   $$\sigma * \tau = \langle 0, i_1, 0, i_2, \ldots, 0, i_m\rangle^\frown\langle 1, j_1, 1, j_2, \ldots, 1, j_n\rangle\ .$$

   Note that from $\sigma * \tau$ we can recover σ and τ. Note also that $|\sigma * \tau| = 2|\sigma| + 2|\tau|$. Now let M be a machine such that

   $$M(\sigma_1 * \sigma_2) \simeq U(\sigma_1)^\frown U(\sigma_2)$$

   for all bitstrings σ_1, σ_2. Let ρ be a "code" for M, i.e. $U(\rho^\frown\sigma) \simeq M(\sigma)$ for all σ. Given τ_1, τ_2 let σ_1, σ_2 be "shortest descriptions" of τ_1, τ_2; $U(\sigma_1) \simeq \tau_1$, $|\sigma_1| = C(\tau_1)$ and $U(\sigma_2) \simeq \tau_2$, $|\sigma_2| = C(\tau_2)$. Then

   $$\begin{array}{rcl} U(\rho^\frown(\sigma_1 * \sigma_2)) & \simeq & M(\sigma_1 * \sigma_2) \\ & \simeq & U(\sigma_1)^\frown U(\sigma_2) \\ & \simeq & \tau_1{}^\frown\tau_2 \end{array}$$

   so

   $$\begin{array}{rcl} C(\tau_1{}^\frown\tau_2) & \leq & |\rho^\frown(\sigma_1 * \sigma_2)| \\ & = & |\rho| + |\sigma_1 * \sigma_2| \\ & = & |\rho| + 2|\sigma_1| + 2|\sigma_2| \\ & = & |\rho| + 2C(\tau_1) + 2C(\tau_2) \\ & = & 2C(\tau_1) + 2C(\tau_2) + O(1). \end{array}$$

   □

32 Partial recursive functionals, etc.

Remark 32.1. The following three spaces are important for us.

1. $\mathbb{N}^\mathbb{N}$ = the *Baire space.*
2. $2^\mathbb{N} = \{0,1\}^\mathbb{N}$ = the *Cantor space.*
3. $\mathbb{N}$ = the natural numbers.

Recall that $\mathbb{N}^\mathbb{N} = \{f : \mathbb{N} \to \mathbb{N}\}$, the space of total 1-place number-theoretic functions. We use letters such as $f, g, h, \ldots$ for points of $\mathbb{N}^\mathbb{N}$. We also consider $2^\mathbb{N}$, the space of all $0,1$-valued total 1-place number-theoretic functions, $2^\mathbb{N} = \{X : \mathbb{N} \to \{0,1\}\}$. Note that $2^\mathbb{N} \subset \mathbb{N}^\mathbb{N}$. We use letters such as $X, Y, Z, \ldots$ to denote points in $2^\mathbb{N}$.

We deal explicitly with the Baire space, but everything that we are saying applies also to the Cantor space as well as various "mixed" spaces.

Definition 32.2. A *partial functional* is a function

$$\Phi : \subseteq \mathbb{N}^{\mathbb{N}} \times \mathbb{N}^k \to \mathbb{N} .$$

Note that $\mathrm{dom}(\Phi)$ = the domain of Φ, a subset of $\mathbb{N}^{\mathbb{N}} \times \mathbb{N}^k$, and $\mathrm{rng}(\Phi)$ = the range of Φ, a subset of $\mathbb{N}$. We use notations such as $\Phi(f, x_1, \ldots, x_k) \simeq y$ and $\Phi(f, x_1, \ldots, x_k) \downarrow$ and $\Phi(f, x_1, \ldots, x_k) \uparrow$ for partial functionals, just as for partial functions. Here f ranges over $\mathbb{N}^{\mathbb{N}}$ and $x_1, \ldots, x_k$ range over $\mathbb{N}$.

Definition 32.3. A partial functional Φ as above is said to be *partial recursive* if and only if

$$\exists e\, (\forall f \in \mathbb{N}^{\mathbb{N}})\, (\forall x_1, \ldots, x_k, y \in \mathbb{N})\, (\Phi(f, x_1, \ldots, x_k) \simeq y \equiv \varphi_e^{(1),f}(x_1, \ldots, x_k) \simeq y)\,.$$

In other words, $\Phi(f, x_1, \ldots, x_k) \simeq \varphi_e^{(1),f}(x_1, \ldots, x_k)$ for all $f, x_1, \ldots, x_k$.

Example 32.4. An example of a partial recursive functional is

$$\Phi(f, x) \simeq \mu y\, (y > x \wedge f(y) = 0).$$

Note that for this Φ we have $\Phi(f, x) \downarrow \equiv \exists y\, (y > x \wedge f(y) = 0)$. Homework #7 Problem 1 is to exhibit an oracle machine computing this partial functional.

Definition 32.5. A predicate $R \subseteq \mathbb{N}^{\mathbb{N}} \times \mathbb{N}^k$ is said to be *recursive* if and only if its characteristic function $\chi_R : \mathbb{N}^{\mathbb{N}} \times \mathbb{N}^k \to \{0, 1\}$ is recursive.

Definition 32.6. For $n \geq 1$, a predicate $P \subseteq \mathbb{N}^{\mathbb{N}} \times \mathbb{N}^k$ is said to be Σ^0_n if

$$P(f, x_1, \ldots, x_k) \equiv \exists y_1\, \forall y_1 \cdots \begin{smallmatrix}\forall \\ \exists\end{smallmatrix}\, y_n\, R(f, x_1, \ldots, x_k, y_1, \ldots, y_n)$$

where R is a recursive predicate. Similarly we extend the definitions of Π^0_n and Δ^0_n for $n \geq 1$ to predicates $P \subseteq \mathbb{N}^{\mathbb{N}} \times \mathbb{N}^k$.

Remark 32.7. All of the usual closure properties hold in the context of predicates $P \subseteq \mathbb{N}^{\mathbb{N}} \times \mathbb{N}^k$. The class of Σ^0_n predicates is closed under $\wedge$, $\vee$, bounded quantification, total recursive substitution, etc. A predicate is Σ^0_n if and only if its negation is Π^0_n, etc. Thus we can perform Tarski/Kuratowski computations in this context.

We now consider how to relativize the above to a fixed Turing oracle g.

Remark 32.8. A useful fact about the spaces $\mathbb{N}^{\mathbb{N}}$ and $2^{\mathbb{N}}$ is the existence of a $1-1$ onto pairing function

$$\begin{array}{ccc} \mathbb{N}^{\mathbb{N}} \times \mathbb{N}^{\mathbb{N}} & \underset{\text{onto}}{\overset{1-1}{\longleftrightarrow}} & \mathbb{N}^{\mathbb{N}} \\ (f, g) & & f \oplus g \end{array}$$

where

$$\begin{aligned} (f \oplus g)(2n) &= f(n) \\ (f \oplus g)(2n+1) &= g(n) \end{aligned}$$

and

$$\begin{array}{ccc} 2^{\mathbb{N}} \times 2^{\mathbb{N}} & \underset{\text{onto}}{\overset{1-1}{\longleftrightarrow}} & 2^{\mathbb{N}} \\ (X,Y) & & X \oplus Y \end{array}$$

as a special case.

Definition 32.9. Let g be a fixed oracle, i.e., $g \in \mathbb{N}^{\mathbb{N}}$. A partial functional $\Phi : \subseteq \mathbb{N}^{\mathbb{N}} \times \mathbb{N}^k \to \mathbb{N}$ is said to be *partial g-recursive*, or *partial recursive relative to g*, if

$$\exists e \, (\forall f \in \mathbb{N}^{\mathbb{N}}) \, (\forall x_1, \ldots, x_k \in \mathbb{N}) \, (\Phi(g, x_1, \ldots, x_k) \simeq \varphi_e^{(k), f \oplus g}(x_1, \ldots, x_k)) \ .$$

Similarly we define what it means for a predicate $P \subseteq \mathbb{N}^{\mathbb{N}} \times \mathbb{N}^k$ to be g-recursive, $\Sigma_n^{0,g}$, etc.

Remark 32.10. We can extend all of this to "mixed" functionals and predicates on spaces such as

$$(\mathbb{N}^{\mathbb{N}})^m \times (2^{\mathbb{N}})^l \times \mathbb{N}^k$$

using the pairing function $\oplus$. For example, a partial functional

$$\Phi : \subseteq (\mathbb{N}^{\mathbb{N}})^m \times (2^{\mathbb{N}})^l \times \mathbb{N}^k \to \mathbb{N}$$

is said to be partial g-recursive if and only if it is defined by

$$\Phi(f_1, \ldots, f_m, X_1, \ldots, X_l, x_1, \ldots, x_k) \simeq \varphi_e^{(k), f_1 \oplus \cdots \oplus f_m \oplus X_1 \oplus \cdots \oplus X_l \oplus g}(x_1, \ldots, x_k)$$

for some fixed e and for all $f_1, \ldots, f_m \in \mathbb{N}^{\mathbb{N}}$, $X_1, \ldots, X_l \in 2^{\mathbb{N}}$, $x_1, \ldots, x_k \in \mathbb{N}$.

Remark 32.11. Post's Theorem fails in this context. For predicates $P \subseteq \mathbb{N}^{\mathbb{N}} \times \mathbb{N}^k$ or $P \subseteq 2^{\mathbb{N}} \times \mathbb{N}^k$, it is *not* true in general that $P \in \Sigma_2^0 \equiv P \in \Sigma_1^{0,0'}$. It is true that $P \in \Sigma_1^{0,0'}$ implies $P \in \Sigma_2^0$, but the converse does not hold in general. See also Homework #8, Problem 2.

33 Homework #8, due October 22, 2007

Exercises 33.1.

1. Let $K(\tau)$ denote the *prefix-free complexity* of a bitstring τ. Prove that

$$K(\tau_1 {}^\frown \tau_2) \quad \leq \quad K(\tau_1) + K(\tau_2) + O(1) \ .$$

2. (a) Give an example of a subset of $\mathbb{N}^{\mathbb{N}}$ which is Σ_2^0 but not $\Sigma_1^{0,0'}$.

   (b) Can you replace $\mathbb{N}^{\mathbb{N}}$ by $2^{\mathbb{N}}$ here?

   Note: Recall Post's Theorem, which says (among other things) that a subset of $\mathbb{N}$ is Σ_2^0 if and only if it is $\Sigma_1^{0,0'}$. The point of (a) is to show that Post's Theorem does not hold for subsets of $\mathbb{N}^{\mathbb{N}}$.

   Hint: Recall that a set is open if and only if it is Σ_1^0 relative to an oracle. Therefore, it suffices to find a set which is Σ_2^0 and not open.

3. A real number is said to be *left recursively enumerable* (respectively *right recursively enumerable*) if it is the limit of an increasing (respectively decreasing) recursive sequence of rational numbers.

   (a) If A is a recursively enumerable subset of $\mathbb{N}$, show that the real number $\sum_{n\in A} 1/2^n$ is left recursively enumerable.

   (b) Show that there exist real numbers which are left recursively enumerable but not recursive.

   (c) Show that a real number is recursive if and only if it is both left recursively enumerable and right recursively enumerable.

4. Let P be a Π^0_1 subset of $2^{\mathbb{N}}$. We have seen how to construct a recursive tree $T \subseteq 2^{<\mathbb{N}}$ such that $P = \{\text{paths through } T\}$. For each $n = 0, 1, 2, \ldots$ let T_n be the set of strings in T of length n.

   (a) Show that T_n is prefix-free.

   (b) Show that the set

   $$V_n \quad = \quad \bigcup_{\tau \in T_n} N_\tau$$

   is Δ^0_1. (Note that V_n is a subset of $2^{\mathbb{N}}$.)

   (c) Show that P is the intersection of the V_n's. In other words,

   $$P \quad = \quad \bigcap_{n=0}^{\infty} V_n \ .$$

   (d) Show that the measure of P is given by

   $$\mu(P) \quad = \quad \lim_{n\to\infty} \frac{|T_n|}{2^n} \ .$$

   Here $|T_n|$ denotes the number of strings in T_n.

   (e) Show that the real number $\mu(P)$ is right recursively enumerable.

   (f) Show that $\mu(P)$ is not necessarily a recursive real number.

5. Given a nonempty Π^0_1 set $P \subseteq 2^{\mathbb{N}}$, can we always find a member of P which is recursive?

   Hint: Consider a recursively inseparable pair of r.e. sets.

6. Two sets $P, Q \subseteq \mathbb{N}^{\mathbb{N}}$ are said to be *Turing isomorphic* if the members of P and Q have the same Turing degrees, i.e.,

   $$\{\deg_T(f) \mid f \in P\} \quad = \quad \{\deg_T(g) \mid g \in Q\} \ .$$

   (a) Prove that every Π^0_2 subset of $\mathbb{N}^{\mathbb{N}}$ is Turing isomorphic to a Π^0_1 subset of $\mathbb{N}^{\mathbb{N}}$.

(b) Prove that every Π^0_2 subset of $\mathbb{N}^\mathbb{N}$ is Turing isomorphic to a Π^0_2 subset of $2^\mathbb{N}$.

(c) Is every Π^0_2 subset of $2^\mathbb{N}$ Turing isomorphic to a Π^0_1 subset of $2^\mathbb{N}$? Justify your answer.

Hints: (a) If $\forall x\, \exists y\, R(f, x, y)$ holds, map f to $f \oplus g$ where $g(x) = \mu y\, R(f, x, y)$. (b) Map f to the characteristic function of the set $G_f = \{3^x 5^y \mid f(x) = y\}$ = the "graph" of f.

Lecture 21: October 15, 2007

34 Σ^0_1 and Π^0_1 sets in $2^\mathbb{N}$ and $\mathbb{N}^\mathbb{N}$

We have defined the three spaces $\mathbb{N}$, $2^\mathbb{N}$, and $\mathbb{N}^\mathbb{N}$ (the natural numbers, the Cantor space, and the Baire space). For mixed predicates, $P \subseteq (\mathbb{N}^\mathbb{N})^m \times (2^\mathbb{N})^l \times \mathbb{N}^k$, we know what it means for $P(-,-,-)$ to be recursive, or Σ^0_n, or Σ^0_n relative to an oracle, etc.

Remark 34.1. All of our rules about combining predicates apply in this context. If $P, Q \in \Sigma^0_n$ then $P \wedge Q \in \Sigma^0_n$, $P \vee Q \in \Sigma^0_n$, $\neg P \in \Pi^0_n$). P is recursive if and only if P is Δ^0_1. The class of Σ^0_n predicates is closed under bounded quantification, recursive substitution, etc.

Remark 34.2. Post's Theorem fails in this context. For instance, we can find a Σ^0_2 set $P \subseteq \mathbb{N}^\mathbb{N}$ which is not $\Sigma^{0,\mathbf{0}'}_1$. See Homework #8, Problem 2.

For the sake of our later discussion of randomness, we want to focus on Σ^0_1 sets and Π^0_1 sets in $2^\mathbb{N}$, the Cantor space.

Recall that

$$\begin{aligned} 2^\mathbb{N} &= \{X \mid \mathbb{N} \to \{0,1\}\} \\ &= \{\text{infinite sequences of 0's and 1's}\}. \end{aligned}$$

Each point $X \in 2^\mathbb{N}$ is an infinite sequence of 0's and 1's, i.e.,

$$X = \langle X(0), X(1), \ldots, X(n), \ldots \rangle$$

with each $X(n)$ = either 0 or 1. We view X as the outcome of an infinite sequence of independent coin tosses using a "fair coin", i.e., probability of heads = probability of tails. We identify 1 as heads, 0 as tails. This corresponds to the *fair coin measure* on the space $2^\mathbb{N}$.

In order to define the fair coin measure rigorously, recall that

$$2^{<\mathbb{N}} = \{\text{bitstrings}\} = \{\text{0,1-valued strings}\}.$$

Definition 34.3. Given a bitstring $\sigma \in 2^{<\mathbb{N}}$ and a point $X \in 2^\mathbb{N}$, write $\sigma \subset X$ to mean that σ is an *initial segment* of X, i.e., $\sigma = \langle X(0), X(1), \ldots, X(n-1) \rangle = X \upharpoonright n$ for some n. We then have $n = |\sigma|$. Define

$$N_\sigma = \{X \in 2^{\mathbb{N}} \mid \sigma \subset X\} = \text{the } \textit{neighborhood} \text{ determined by } \sigma.$$

The *fair coin probability measure* is defined as the unique measure μ on $2^{\mathbb{N}}$ with $\mu(N_\sigma) = 1/2^{|\sigma|}$ for all bitstrings σ. To motivate this definition, note that if $|\sigma| = n$ then the probability of the event $X \upharpoonright n = \sigma$ is $1/2^n = 1/2^{|\sigma|}$.

Definition 34.4. A set $U \subseteq 2^{\mathbb{N}}$ is said to be *open* if U is the union of a collection of neighborhoods. In other words,

$$U = \bigcup_{\sigma \in S} N_\sigma$$

where $S \subseteq 2^{<\mathbb{N}}$ is a set of bitstrings.

Remark 34.5. Σ^0_1 sets in the Cantor space are open. This is because of finite approximation. Let us elaborate.

A typical Σ^0_1 set in $2^{\mathbb{N}}$ is

$$U_e = \{X \in 2^{\mathbb{N}} \mid \varphi_e^{(1),X}(0)\downarrow\} .$$

We use U_e, $e = 0, 1, 2, \ldots$ as our standard recursive enumeration of the Σ^0_1 sets in $2^{\mathbb{N}}$. By finite approximation, $X \in U_e \equiv \exists\sigma\,(\sigma \subset X \wedge \varphi_e^{(1),\sigma}(0)\downarrow)$. This shows that $X \in U_e$ depends only on a finite amount of information from X, i.e., it depends only on an initial segment of X.

Recall the notation $\varphi_{e,s}^{(1),\sigma}(x) \simeq y$ which means that the oracle computation with input x and oracle f halts in $\leq s$ steps with output y using only oracle information from $\sigma \subset f$. This predicate is recursive. Furthermore, the predicate

$$\varphi_e^{(1),\sigma}(x) \simeq \varphi_{e,|\sigma|}^{(1),\sigma}(x)$$

is also recursive. We then have

$$U_e = \bigcup_{\varphi_e^{(1),\sigma}(0)\downarrow} N_\sigma .$$

Hence U_e is an open set.

We now refine the above remark to get a useful technical fact. Note that for each $X \in U_e$ there is a unique shortest initial segment $\sigma \subset X$ such that $\varphi_e^{(1),\sigma}(0)\downarrow$. Thus we have

$$\begin{aligned} X \in U_e &\equiv \exists\sigma\,\Big(\sigma \subset X \wedge \varphi_e^{(1),\sigma}(0)\downarrow\Big) \\ &\equiv \exists\sigma\,\Big(\sigma \subset X \wedge \varphi_e^{(1),\sigma}(0)\downarrow \,\wedge \neg\,\exists\tau \subset \sigma\,(\varphi_e^{(1),\tau}(0)\downarrow)\Big) \\ &\equiv X \in \bigcup_{\sigma \in S_e} N_\sigma \end{aligned}$$

where S_e is a recursive set of bitstrings, namely

$$S_e = \{\sigma \mid \varphi_e^{(1),\sigma}(0)\downarrow \,\wedge \neg\,\exists\tau \subset \sigma\,(\varphi_e^{(1),\tau}(0)\downarrow)\} .$$

Note also that S_e is *prefix-free*, in the following sense.

Definition 34.6. If $\sigma, \tau \in 2^{<\mathbb{N}}$ are bitstrings, $\sigma \subseteq \tau$ means that σ is an *initial segment* of τ (possibly $\sigma = \tau$). Also $\sigma \subset \tau$ means that σ is a *proper initial segment* or *prefix* of τ, i.e., $\sigma \subseteq \tau$ and $\sigma \neq \tau$. A set of bitstrings $S \subseteq 2^{<\mathbb{N}}$ is said to be *prefix-free* if $\neg\,(\exists \sigma \in S)\,(\exists \tau \in S)\,(\sigma \subset \tau)$.

Summarizing, we have proved the following theorem which says among other things that Σ^0_1 sets are open.

Theorem 34.7. For a set $U \subseteq 2^{\mathbb{N}}$, the following are pairwise equivalent.

1. U is Σ^0_1 .
2. $U = U_e$ for some e.
3. $U = \bigcup_{\sigma \in S} N_\sigma$ for some recursively enumerable set of bitstrings S.
4. $U = \bigcup_{\sigma \in S} N_\sigma$ for some recursive, prefix-free set of bitstrings S.

Proof. $1 \Leftrightarrow 2$ by definition.
$2 \Rightarrow 4$ is what we have already proved.
$4 \Rightarrow 3$ is trivial.
$3 \Rightarrow 1$: Assuming 3, we have

$$\begin{aligned} X \in U \quad &\equiv \quad X \in \textstyle\bigcup_{\sigma \in S} N_\sigma, S \text{ is r.e.} \\ &\equiv \quad \underbrace{\exists \sigma \underbrace{\left(\underbrace{\sigma \in S}_{\Sigma^0_1} \wedge \underbrace{\sigma \subset X}_{\text{recursive}} \right)}_{\Sigma^0_1}}_{\Sigma^0_1} \end{aligned}$$

which proves 1. □

Remark 34.8. If an open set has been written as

$$U = \bigcup_{\sigma \in S} N_\sigma$$

where S is prefix-free, then we can find the measure of U as follows. Note first that $\sigma \subseteq \tau$ if and only if $N_\sigma \supseteq N_\tau$. On the other hand, if $\sigma \nsubseteq \tau$ and $\tau \nsubseteq \sigma$ (i.e., σ is *incompatible* with τ, abbreviated $\sigma \,|\, \tau$), then $N_\sigma \cap N_\tau = \emptyset$. Thus, for any prefix-free set of bitstrings S, $U = \bigcup_{\sigma \in S} N_\sigma$ is a union of pairwise disjoint neighborhoods, hence

$$\mu(U) = \sum_{\sigma \in S} \frac{1}{2^{|\sigma|}} \ .$$

We also have:

Theorem 34.9. For a set $U \subseteq 2^{\mathbb{N}}$, the following are pairwise equivalent.

1. U is open.
2. U is $\Sigma_1^{0,f}$ for some oracle f.
3. $U = \bigcup_{\sigma \in S} N_\sigma$ for some set of bitstrings S.
4. $U = \bigcup_{\sigma \in S} N_\sigma$ for some prefix-free set of bitstrings S.

   In the latter case we have $\mu(U) = \sum_{\sigma \in S} 1/2^{|\sigma|}$.

Proof. 1 $\Leftrightarrow$ 3 holds by definition. If 3 holds then clearly U is $\Sigma_1^{0,S}$, hence 2 holds. The previous theorem relativizes to prove 2 $\Rightarrow$ 4. The implication 4 $\Rightarrow$ 3 is trivial. This completes the proof. □

Lecture 22: October 17, 2007

Review: We have seen that a typical Σ_1^0 set in the Cantor space $2^{\mathbb{N}}$ looks like

$$U_e = \{X \in 2^{\mathbb{N}} \mid \varphi_e^{(1),X}(0)\downarrow\}.$$

We have proved that

$$U_e = \bigcup_{\sigma \in S_e} N_\sigma$$

where S_e is a set of bitstrings which is recursive and *prefix-free*, i.e., $\neg\,(\exists \sigma, \tau \in S_e)\,(\sigma \subset \tau)$. Here the neighborhoods are defined by $N_\sigma = \{X \in 2^{\mathbb{N}} \mid \sigma \subset X\}$ for all bitstrings σ.

Remark 34.10. The same analysis holds for Σ_1^0 sets in the Baire space, $\mathbb{N}^{\mathbb{N}}$, using strings instead of bitstrings. In this case we would have to define the neighborhoods differently, namely we would have $N_\sigma = \{f \in \mathbb{N}^{\mathbb{N}} \mid \sigma \subset f\}$ for all strings σ.

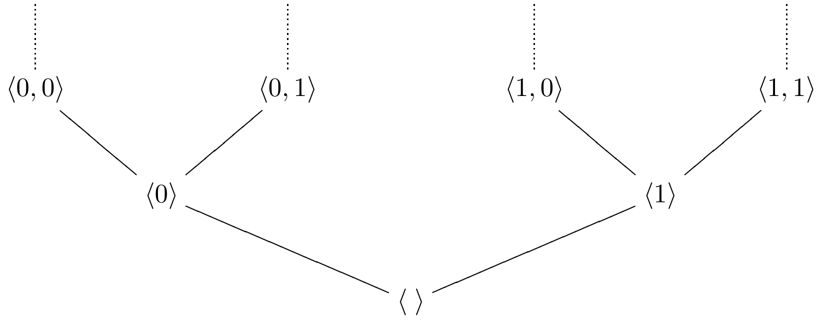
We now consider the structure of Π_1^0 sets.

Of course, $P \subseteq \mathbb{N}^{\mathbb{N}}$ is Π_1^0 if and only if $\mathbb{N}^{\mathbb{N}} \setminus P$ is Σ_1^0. Hence, a picture of a Π_1^0 set can be obtained by viewing it as the complement of a Σ_1^0 set, which in turn is described by a recursive, prefix-free set of strings.

We wish to observe that another useful picture of Π_1^0 sets can be obtained in terms of *trees*. See also Homework #8, Problem 4.

Definition 34.11. A *tree* is a set of strings, T, which is closed when taking initial segments. In other words, $\forall \sigma\, \forall \tau\, (\sigma \subseteq \tau, \tau \in T \Rightarrow \sigma \in T)$.

Example 34.12. $T = 2^{<\mathbb{N}}$ = the full binary tree.



Remark 34.13. Trees are in a sense the opposite of prefix-free sets of strings. T is a tree if and only if all prefixes of members of T are members of T. S is prefix-free if and only if no prefix of a member of S is a member of S.

Definition 34.14. Let T be a tree. A *path through* T is a function $f \in \mathbb{N}^{\mathbb{N}}$ such that $f \upharpoonright n \in T$ for all n.

Example 34.15. The Π^0_1 set $2^{\mathbb{N}} \subseteq \mathbb{N}^{\mathbb{N}}$ is the set of paths through $2^{<\mathbb{N}}$, the full binary tree.

Theorem 34.16. A set $P \subseteq \mathbb{N}^{\mathbb{N}}$ is Π^0_1 if and only if $P = \{\text{paths through } T\}$ for some recursive tree T.

Proof. ($\Rightarrow$) Assume P is Π^0_1. Then

$$\begin{aligned} P &= \{f \in \mathbb{N}^{\mathbb{N}} \mid \varphi_e^{(1),f}(0)\uparrow\} \\ &= \{f \in \mathbb{N}^{\mathbb{N}} \mid \forall n\ \varphi_e^{(1),f\upharpoonright n}(0)\uparrow\} \\ &\qquad \text{(by finite approximation)} \\ &= \{f \in \mathbb{N}^{\mathbb{N}} \mid f \text{ is a path through } T\} \end{aligned}$$

where $T = \{\tau \in \mathbb{N}^{<\mathbb{N}} \mid \varphi_e^{(1),\tau}(0)\uparrow\}$. Note that T is a recursive tree.

($\Leftarrow$) Assume $P = \{\text{paths through } T\}$ for some recursive tree T. Then

$$\begin{aligned} P &= \{f \in \mathbb{N}^{\mathbb{N}} \mid \forall n\ \underbrace{f \upharpoonright n \in T}_{\text{recursive}}\} \\ &= \{f \in \mathbb{N}^{\mathbb{N}} \mid \forall n\, R(f,n)\} \\ &= \Pi^0_1 \end{aligned}$$

□

Remark 34.17. The same applies to Π^0_1 sets in the Cantor space, $2^{\mathbb{N}}$. Note also that if $P \subseteq 2^{\mathbb{N}}$ is Π^0_1 then we may restrict our attention to bitstrings, so we may take our recursive tree T to be a subtree of $2^{<\mathbb{N}}$.

Definition 34.18. A set $P \subseteq \mathbb{N}^{\mathbb{N}}$ is defined to be *closed* if it complement $\mathbb{N}^{\mathbb{N}} \setminus P$ is open.

In analogy with Theorems 34.9 and 34.16 we have

Theorem 34.19. For $P \subseteq \mathbb{N}^{\mathbb{N}}$ the following are pairwise equivalent.

1. P is closed.
2. P is $\Pi^{0,f}_1$ for some oracle f.
3. $P = \{\text{paths through } T\}$ for some tree T.

Moreover, if $P \subseteq 2^{\mathbb{N}}$ then we may take T to be a subtree of $2^{<\mathbb{N}}$.

Proof. This follows easily from Theorems 34.9 and 34.16 and relativization. □

35 Prefix-free complexity

In this section we consider a variant of Kolmogorov complexity which is somewhat better behaved.

Recall that a *machine* is a partial recursive function from bitstrings to bitstrings,

$$M :\subseteq 2^{<\mathbb{N}} \to 2^{<\mathbb{N}} .$$

It follows that $\mathrm{dom}(M)$ is a recursively enumerable set of bitstrings.

Definition 35.1. A *prefix-free machine* is a machine M such that $\mathrm{dom}(M)$ is prefix-free.

Intuitively, $\mathrm{dom}(M)$ consists of "self-delimiting programs," i.e., each "valid program" $\sigma \in \mathrm{dom}(M)$ has an "end-of-program marker" and therefore cannot be an initial segment of another program in $\mathrm{dom}(M)$.

Definition 35.2. A *universal prefix-free machine* is a prefix-free machine, call it U, such that

$$(\forall \text{ prefix-free machines } M)\,(\exists \text{ bitstring } \rho)\,(\forall \text{ bitstrings } \sigma)\,(U(\rho^\frown\sigma) \simeq M(\sigma)).$$

Theorem 35.3. Universal prefix-free machines exist.

Proof. Later. □

Definition 35.4. Let U be a fixed universal prefix-free machine. The *prefix-free complexity* of a bitstring τ is defined as

$$K(\tau) \;=\; \min\{|\sigma| \mid U(\sigma) \simeq \tau\} .$$

Remark 35.5. Prefix-free complexity, $K(\tau)$, has the same nice properties as "plain" complexity, $C(\tau)$. In fact, we shall see that it tends to have even better properties.

Theorem 35.6. $K(\tau)$ is well-defined up to an additive constant.

The precise statement reads as follows. If U_1 and U_2 are two universal prefix-free machines, and if for $i = 1, 2$ we define $K_i(\tau) = \min\{|\sigma| \mid U_i(\sigma) \simeq \tau\}$, then $|K_1(\tau) - K_2(\tau)| \leq O(1)$ for all τ.

Proof. As before. □

Also as before define $K(n) = K(\langle\underbrace{0, \ldots, 0}_{n}\rangle)$. We then have:

Theorem 35.7.

1. $K(|\tau|) \leq K(\tau) + O(1)$.
2. $K(\tau) \leq C(\tau) + K(|\tau|) + O(1)$. In particular $K(\tau) \leq |\tau| + 2\log_2 |\tau| + O(1)$, etc.
3. $K(\tau_1{}^\frown\tau_2) \leq K(\tau_1) + K(\tau_2) + O(1)$.

Note that in the case of "plain" complexity there was an annoying factor of 2 in part 3. For prefix-free complexity this factor of 2 disappears and is not needed.

Proof. Parts 1 and 2 are proved as before. Part 3 is Homework #8, Problem 1. □

Remark 35.8. It would be interesting to compare $C(\tau)$ and $K(\tau)$. For instance, it is easy to see that $C(\tau) \leq K(\tau) + O(1)$ and $K(\tau) \leq 2C(\tau) + O(1)$. To what extent can we improve these inequalities? This would be an interesting research project.

We now prove Theorem 35.3 by constructing a universal prefix-free machine. As before let M_e, $e = 0, 1, 2, \ldots$ be our standard recursive enumeration of all machines, given by

$$M_e(\sigma) \simeq \tau \quad \equiv \quad \varphi_e^{(1)}(\#(\sigma)) \simeq \#(\tau)$$

where σ and τ are bitstrings. Define

$$M_{e,s}(\sigma) \simeq \tau \quad \equiv \quad \#(\sigma) < s \text{ and } \varphi_{e,s}^{(1)}(\#(\sigma)) \simeq \#(\tau)\ .$$

Some easy facts are:

1. $M_e(\sigma) \simeq \tau$ if and only if $\exists s\, (M_{e,s}(\sigma) \simeq \tau)$.
2. If $s \leq t$ and $M_{e,s}(\sigma) \simeq \tau$ then $M_{e,t}(\sigma) \simeq \tau$.
3. The 4-place predicate $M_{e,s}(\sigma) \simeq \tau$ is recursive.
4. The 3-place predicate $M_{e,s}(\sigma)\downarrow$ is recursive.
5. For all $\sigma \in \text{dom}(M_{e,s})$ we have $\#(\sigma) < s$.

Define $\widetilde{M}_e$ to be the obvious "prefix-free restriction" of M_e, namely

$$\widetilde{M}_e(\sigma) \simeq \tau \quad \equiv \quad \exists s\ [\underbrace{M_{e,s}(\sigma) \simeq \tau}_{\text{recursive}} \text{ and } \underbrace{\text{dom}(M_{e,s}) \text{ is prefix-free}}_{\text{recursive}}]\,,$$

and note that $\widetilde{M}_e$ is partial recursive. Some easy facts are:

1. For all e, $\widetilde{M}_e$ is a prefix-free machine.
2. For each e, if M_e is a prefix-free machine then $\widetilde{M}_e = M_e$.
3. $\widetilde{M}_e(\sigma)$ is a partial recursive function of e and σ.

Thus we see that $\widetilde{M}_e$, $e = 0, 1, 2, \ldots$ is a recursive enumeration of all prefix-free machines. It follows that the machine $\widetilde{U}$ defined by

$$\widetilde{U}(\langle\underbrace{0, \ldots, 0}_{e}, 1\rangle^\frown\sigma) \quad \simeq \quad \widetilde{M}_e(\sigma)$$

is a universal prefix-free machine.

Lecture 23: October 19, 2007

36 Foundations of mathematics

The purpose of foundations of mathematics is to understand very clearly and precisely the most basic concepts of mathematics. We wish to answer questions such as:

1. What is a number?
2. What is a shape?
3. What is a set?
4. What is a function?
5. What is an axiom?
6. What is a theorem?
7. What is a proof?
8. What is an algorithm?

All of mathematics is built on these concepts, yet often we proceed without having a precise idea of what they mean. There is a lot of interest in these questions, and the issue of how these questions should be answered is an important topic in philosophy of mathematics.

Researchers in foundations of mathematics have made a lot of progress on these question. Concepts such as set, function, and number have been grounded and made precise in set theory. Concepts such as theorem, proof, and axiom have been made precise in mathematical logic.

One example we have seen of defining a fundamental concept is Turing's definition of a computable function, which is widely recognized as the "right definition", i.e., the right answer to the question

> What is a computable function?

Turing's definition has intuitive appeal. The functions we believe should be computable satisfy the definition, and conversely. Before Turing, the idea of a computable function had not been made precise.

In a similar vein, we now wish to clarify the concept of randomness, i.e., to answer the question

> What is a *random* point in a probability space?

37 Definitions of randomness

We wish to define what we mean by a random point in a probability space.

To keep things simple, we consider only one probability space: the Cantor space, $2^{\mathbb{N}}$, with the fair coin probability measure, μ. Recall that each bitstring $\sigma \in 2^{<\mathbb{N}}$ determines a neighborhood $N_\sigma = \{X \in 2^{\mathbb{N}} \mid \sigma \subset X\}$ in $2^{\mathbb{N}}$. The measure of N_σ is $\mu(N_\sigma) = 1/2^{|\sigma|} = \text{Prob}(\sigma \text{ is an initial segment of } X)$. The measure μ has the following properties:

1. $\mu(2^{\mathbb{N}}) = 1$ and $\mu(\emptyset) = 0$.
2. $\mu(\bigcup_{i=0}^{\infty} S_i) = \sum_{i=0}^{\infty} \mu(S_i)$ provided the sets S_i, $i = 0, 1, 2, \ldots$ are pairwise disjoint.
3. $\mu(2^{\mathbb{N}} \setminus S) = 1 - \mu(S)$.

We wish to define what we might mean by saying that a point $X \in 2^{\mathbb{N}}$, $X =$ an infinite sequence of 0's and 1's, is *random*. Our first attempt is as follows.

Definition 37.1 (non-rigorous). A point $X \in 2^{\mathbb{N}}$ is said to be *random* if it is the outcome of an infinite sequence of tosses of an unbiased coin, identifying heads as 1 and tails as 0.

This definition, although not rigorous, provides some guidance. For example, we would not expect a random X to have $X(2n) = 1$ for all n, because this would mean that all of the even-numbered coin tosses result in heads, an event which is highly unlikely. This corresponds to the fact that

$$\text{Prob}(\forall n\,(X(2n) = 1)) = \mu\{X \in 2^{\mathbb{N}} \mid \forall n\,(X(2n) = 1)\} = 0.$$

These considerations suggest the following attempt at defining randomness, using simple concepts from measure theory.

Definition 37.2 (temporary). A point $X \in 2^{\mathbb{N}}$ is said to be *random* if X does not belong to any set $S \subseteq 2^{\mathbb{N}}$ which is of measure 0.

This definition of randomness has the advantage of being perfectly rigorous. Furthermore, if X is random according to this definition, then X obviously has many properties which are intuitively associated with sequences of coin tosses. For instance, if X is random then $\neg \forall n\, (X(2n) = 1)$ as desired. In fact, if X is random, then no event of probability 0 occurs.

A fatal difficulty with this definition is that, under this definition, random points do not exist! For any point $X \in 2^{\mathbb{N}}$, the singleton set $\{X\}$ is a null set, and $X \in \{X\}$, so X is not random. Thus, the above definition of randomness turns out to be uninteresting.

We therefore discard the above definition and replace it be another definition in which we consider only "nice" sets of measure 0, instead of arbitrary sets of measure 0. In order to define what we mean by a "nice" set of measure 0, we use concepts from recursion theory.

Definition 37.3. For each $n \geq 1$, a point $X \in 2^{\mathbb{N}}$ is said to be *weakly n-random* if $X \notin$ any Π^0_n set of measure 0. Equivalently, $X \notin$ any Σ^0_{n+1} set of measure 0.

Remark 37.4. Note that $X \notin$ any Π^0_n set of measure 0 if and only if $X \notin$ any Σ^0_{n+1} set of measure 0. This is because a Σ^0_{n+1} set is a union of Π^0_n sets.

Remark 37.5. In the previous definition, we do not consider Σ^0_1 sets of measure 0. The reason for this restriction is that the only Σ^0_1 set of measure 0 is the empty set. If we were to make the above definition with Σ^0_1 instead of Π^0_n, then all points of $2^{\mathbb{N}}$ would be random, so the definition would be uninteresting.

Lemma 37.6. There exist points $X \in 2^{\mathbb{N}}$ which are weakly n-random.

Proof. There are only countably many Π^0_n sets. Hence, there are only countably many Π^0_n sets of measure 0. Let $S_n =$ the union of all Π^0_n sets of measure 0. Equivalently, S_n is the union of all Σ^0_{n+1} sets of measure 0. By countable additivity, $\mu(S_n) = 0$. Any $X \notin S_n$ is weakly n-random. □

Remark 37.7. Note that S_n itself is not Π^0_n or even Σ^0_{n+1}.

Remark 37.8. It is clear that, for each $n \geq 1$, weak $n+1$-randomness implies weak n-randomness. We shall see later that the converse does not hold.

Definition 37.9. A point $X \in 2^{\mathbb{N}}$ is said to be *arithmetically random* if $X \notin$ any Π^0_n set of measure 0 for any $n \geq 1$. Equivalently, X is weakly n-random for all $n \geq 1$.

Remark 37.10. They exist because, by countable additivity, $\mu(\bigcup_{n=1}^{\infty} S_n) = 0$.

Remark 37.11. If X is weakly 1-random, then X has at least some of the desirable properties which we would normally expect of sequences of coin tosses. For example, $\neg \forall n\, (X(2n) = 1)$. This is because the set $\{X \in 2^{\mathbb{N}} \mid \forall n\, (X(2n) = 1)\}$ is Π^0_1 of measure 0.

Similarly we can show that if X is weakly 1-random then X is the characteristic function of a biimmune set. (A set $B \subseteq \mathbb{N}$ is said to be *biimmune* if both B and the complement of B are immune.) See Homework #9 Problem 6. This gives an example of two immune sets whose union is not immune. Another example can be constructed using finite approximation, but the present example in terms of weak 1-randomness is perhaps easier and more interesting.

Remark 37.12. From the previous remark, we see that the concept of weak 1-randomness is useful. However, we shall see later that this concept is not really what we want. For example, it is possible for X to be weakly 1-random yet not obey the Strong Law of Large Numbers.

A better concept of randomness is due to P. Martin-Löf in a paper published in 1966. We shall see that Martin-Löf's concept of randomness is intermediate between weak 1-randomness and weak 2-randomness and implies essentially all desirable statistical properties which would normally be expected of an infinite sequence of coin tosses.

Our goal now is to present Martin-Löf's definition of randomness. First we need some preliminary definitions.

Definition 37.13. A *null set* is a set $S \subseteq 2^{\mathbb{N}}$ which is of measure 0, i.e., $\mu(S) = 0$.

Remark 37.14. A well known fact is that S is null if and only if

$$(\forall \epsilon > 0)\,(\exists \text{ open set } V)\,(S \subseteq V \wedge \mu(V) < \epsilon)\,.$$

This follows from the fact that the fair coin probability measure μ is *regular.*

Next we are going to "effectivize" the concept of a null set. This means that we are going to define a more restricted concept which pays more attention to computability.

Definition 37.15.

1. A set $V \subseteq 2^{\mathbb{N}}$ is said to be *effectively open* if V is Σ^0_1. (See also Theorem 34.9.)

2. A set $S \subseteq 2^{\mathbb{N}}$ is said to be *effectively null* if there exists a recursive sequence of effectively open sets V_n, $n = 0, 1, 2, \ldots$, such that

$$\forall n\,(S \subseteq V_n \wedge \mu(V_n) \leq 1/2^n)\,.$$

Remark 37.16. Recall that our standard recursive enumeration of all Σ^0_1 subsets of $2^{\mathbb{N}}$ is given by

$$U_e = \{X \in 2^{\mathbb{N}} \mid \varphi_e^{(1),X}(0)\downarrow\}$$

where $e = 0, 1, 2, \ldots$. By a *recursive sequence of effectively open sets* we mean a sequence $V_n = U_{f(n)}$, $n = 0, 1, 2, \ldots$, where $f(n)$ is a total recursive function. In this case we could also say that the sequence V_n, $n = 0, 1, 2, \ldots$ is *uniformly effectively open*, or we could say that the sequence is *uniformly* Σ^0_1.

Remark 37.17. Obviously every effectively null set is a null set, but the converse does not hold. Later we shall prove the surprising result that the union of all effectively null sets is an effectively null set. Thus, there is a unique largest effectively null set.

Finally we are able to present Martin-Löf's definition of randomness.

Definition 37.18 (Martin-Löf). A point $X \in 2^{\mathbb{N}}$ is said to be *random* if $X \notin$ any effectively null set. Equivalently, the singleton set $\{X\}$ is not effectively null.

Exercise 37.19. Let $X \in 2^{\mathbb{N}}$ be random in the sense of Martin-Löf. Prove that for all bitstrings σ there exist infinitely many n such that $X(n+i) = \sigma(i)$ for all $i < |\sigma|$.

Hint: Given a bitstring σ, construct a Martin-Löf test appropriate for σ. In other words, construct an effectively null set S_σ such that every $X \notin S_\sigma$ has the desired property.

38 Homework #9, due October 29, 2007

Exercises 38.1.

1. Hoeffding's Inequality says that the probability space $2^{\mathbb{N}}$ with the fair coin probability measure satisfies

$$\operatorname{Prob}\left(\left|\frac{\sum_{i=0}^{n-1} X(i)}{n} - \frac{1}{2}\right| > \epsilon\right) \quad < \quad \frac{2}{\exp 2n\epsilon^2}\ .$$

   Use Hoeffding's Inequality to prove that if a point $X \in 2^{\mathbb{N}}$ is *random* (i.e., random in the sense of Martin-Löf), then X obeys the Strong Law of Large Numbers:

$$\frac{\sum_{i=0}^{n-1} X(i)}{n} \to \frac{1}{2} \qquad \text{as} \qquad n \to \infty\,.$$

2. Prove that there exist weakly 1-random points in $2^{\mathbb{N}}$ which do not obey the Strong Law of Large Numbers.

   Hint: Use finite approximation.

3. In problem 1, can you say anything about the rate of convergence to $1/2$?

4. Prove that if $X \oplus Y \in 2^{\mathbb{N}}$ is *random* (i.e., random in the sense of Martin-Löf), then $X \not\leq_T Y$ and $Y \not\leq_T X$.

5. Prove that there exist points $X, Y \in 2^{\mathbb{N}}$ such that $X \oplus Y$ is weakly 1-random yet $X \equiv_T Y$.

6. A set $B \subseteq \mathbb{N}$ is said to be *biimmune* if both B and its complement $\mathbb{N} \setminus B$ are immune. Prove that if $X \in 2^{\mathbb{N}}$ is weakly 1-random then X is the characteristic function of a biimmune set.

7. Let f be a Turing oracle.

   For each $i \in \mathbb{N}$ define

   $$U_i^f = \{X \in 2^{\mathbb{N}} \mid \varphi_i^{(1),f \oplus X}(0) \downarrow\}.$$

   Thus U_i^f, $i = 0, 1, 2, \ldots$ is the standard recursive enumeration of all $\Sigma_1^{0,f}$ subsets of $2^{\mathbb{N}}$.

   Given a sequence of sets $V_n \subseteq 2^{\mathbb{N}}$, $n = 0, 1, 2, \ldots$, prove that the following are pairwise equivalent.

   (a) There exists a total recursive function g such that $V_n = U_{g(n)}^f$ for all n.

   (b) There exists a total f-recursive function h such that $V_n = U_{h(n)}^f$ for all n.

   (c) The predicate $P \subseteq 2^{\mathbb{N}} \times \mathbb{N}$ given by

   $$P(X, n) \equiv X \in V_n$$

   is $\Sigma_1^{0,f}$.

   In this case we say that the sequence of sets V_n, $n = 0, 1, 2, \ldots$ is *uniformly* $\Sigma_1^{0,f}$ or *uniformly* Σ_1^0 *relative to* f.

   Note: This concept will be part of the definition of what it means for a point $X \in 2^{\mathbb{N}}$ to be random relative to the oracle f.

Lecture 24: October 22, 2007

39 Properties of Martin-Löf randomness

Review:

We defined a set $S \subseteq 2^{\mathbb{N}}$ to be *effectively null* if $S \subseteq \bigcap_{n=0}^{\infty} V_n$ where V_n, $n = 0, 1, 2, \ldots$ is uniformly Σ_1^0 and $\mu(V_n) \leq 1/2^n$. We defined a point $X \in 2^{\mathbb{N}}$ to be *random* (in the sense of Martin-Löf) if $X \notin$ any effectively null set.

Remark 39.1. Homework #9 Problems 1, 3, and 4 show that if X is random then X has various desirable properties which we would attribute to a sequence of coin tosses:

1. SLLN = Strong Law of Large Numbers:

   $$\lim_{n \to \infty} \frac{\sum_{i=0}^{n-1} X(i)}{n} = \frac{1}{2} .$$

   In other words, "the proportion of heads in the first n coin tosses goes to 1/2 as n goes to infinity."

2. For $X \in 2^{\mathbb{N}}$ we can write X uniquely as $X = X_0 \oplus X_1$ where X_0 and X_1 are the even and odd parts of X, i.e., $X_0(n) = X(2n)$ and $X_1 = X(2n+1)$ for all n. Then, X random implies $X_0 \not\leq_T X_1$ and $X_1 \not\leq_T X_0$. In other words, "the even part does not help us to compute the odd part, and vice versa."

Results such as these tend to justify the Martin-Löf definition of randomness.

We now compare Martin-Löf's concept of randomness to weak n-randomness, $n = 1, 2, \ldots$.

Lemma 39.2. Let P be a subset of $2^{\mathbb{N}}$.

1. If P is Π^0_1 and null, then P is effectively null.
2. If P is effectively null, then $P \subseteq S$ for some S which is Π^0_2 and effectively null.

Proof. 1. If P is Π^0_1, we can write $P = \{\text{paths through } T\}$ where T is a recursive tree. Then

$$P = \bigcap_{n=0}^{\infty} V_n, \quad \text{where } V_n = \bigcup_{\substack{\tau \in T \\ |\tau| = n}} N_\tau$$

with $V_0 \supseteq V_1 \supseteq V_2 \supseteq \cdots \supseteq V_n \supseteq \cdots$ and V_n, $n = 0, 1, 2, \ldots$ is uniformly Σ^0_1. (In fact, V_n is uniformly Δ^0_1.) Hence $\mu(P) = \lim_{n\to\infty} \mu(V_n)$. Clearly the function $\mu(V_n)$ is a recursive function of n, because

$$\mu(V_n) = \sum_{\substack{\tau \in T \\ |\tau| = n}} \frac{1}{2^n} \,.$$

See also Homework #8 Problem 4. If P is null, then

$$\mu(P) = \lim_{n\to\infty} \mu(V_n) = 0$$

so let $f(k) =$ the least n such that $\mu(V_n) \leq 1/2^k$. Then f is a recursive function, so the sets $V^*_k = V_{f(k)}$, $k = 0, 1, 2, \ldots$ are uniformly Σ^0_1 and $\mu(V^*_k) \leq 1/2^k$. We now see that

$$P = \bigcap_{n=0}^{\infty} V_n = \bigcap_{k=0}^{\infty} V^*_k$$

is an effectively null set.

2. If P is effectively null, we have $P \subseteq \bigcap_{n=0}^{\infty} V_n$, $\mu(V_n) \leq 1/2^n$, V_n uniformly Σ^0_1. Letting $S = \bigcap_{n=0}^{\infty} V_n$ we see that $P \subseteq S$, S is effectively null, and

$$X \in S \;\equiv\; \underbrace{\forall n\, (\underbrace{X \in V_n}_{\Sigma^0_1})}_{\Pi^0_2}$$

so S is Π^0_2.

□

Theorem 39.3. Let $X \in 2^{\mathbb{N}}$.

1. X random $\Rightarrow$ X weakly 1-random.
2. X weakly 2-random $\Rightarrow$ X random.

Proof. 1. Suppose X is random. To show X is weakly 1-random, consider any null Π^0_1 set P. By part 1 of Lemma 39.2, P is effectively null. It follows that $X \notin P$.

2. Assume that X is weakly 2-random, i.e., $X \notin$ any null Π^0_2 set. It follows by part 2 of Lemma 39.2 that $X \notin$ any effectively null set, i.e., X is random. □

Remark 39.4. From now on we shall write

random $\equiv$ random in the sense of Martin-Löf,

weakly random $\equiv$ weakly 1-random,

strongly random $\equiv$ weakly 2-random.

The previous theorem tells us that strongly random $\Rightarrow$ random, and random $\Rightarrow$ weakly random. We shall see later that the converses do not hold.

Remark 39.5. Random $\Rightarrow$ not recursive. In fact, weakly random $\Rightarrow$ not recursive. To see this, note that for all $X \in 2^{\mathbb{N}}$ we have $\{X\} = P = \{$paths through $T\}$ where $T = \{X \upharpoonright n \mid n \in \mathbb{N}\}$ is a tree and P is a null set. If X is recursive, then T is recursive, hence P is a Π^0_1 null set, hence X is not weakly random.

An important technical lemma is:

Lemma 39.6 (Solovay's Lemma). Let $X \in 2^{\mathbb{N}}$ be random. Let V_n, $n = 0, 1, 2, \ldots$ be uniformly Σ^0_1 subsets of $2^{\mathbb{N}}$ such that

$$\sum_{n=0}^{\infty} \mu(V_n) \;<\; \infty \;.$$

Then $X \in V_n$ for only finitely many n. In other words, $X \notin V_n$ for all sufficiently large n.

Note: This lemma may be useful in Homework #9, Problem 1.

Lecture 25: October 24, 2007

Proof. By assumption, let c be a constant such that

$$\sum_{n=0}^{\infty} \mu(V_n) \;\leq\; 2^c \;<\; \infty \; .$$

For $k = 0, 1, 2, \ldots$ consider the sets

$$W_k = \{X \in 2^{\mathbb{N}} \mid X \in V_n \text{ for at least } k \text{ many } n\text{'s}\}$$

and note that these sets are uniformly Σ^0_1:

$$X \in W_k \equiv \exists n_1 < \cdots < n_k \quad \underbrace{\underbrace{\forall i \leq k}_{\substack{\text{bounded}\\ \text{quantification}}} \quad (\underbrace{X \in V_{n_i}}_{\Sigma^0_1})}_{\Sigma^0_1}$$

We claim that $\mu(W_k) \leq 2^c/k$ for all k.

Assuming this claim, we have $\mu(W_{2^{c+k}}) \leq 2^c/2^{c+k} = 1/2^k$ and these sets are also uniformly Σ^0_1. Therefore, since X is random, $X \notin W_{2^{c+k}}$ for some k. It follows that $X \in V_n$ for $< 2^{c+k}$ many n's. This proves Solovay's Lemma.

It remains to prove the claim. We have

$$W_k = \{X \in 2^{\mathbb{N}} \mid (\exists^{\geq k} n)\,(X \in V_n)\}$$

so for all s let

$$W_{k,s} = \{X \in 2^{\mathbb{N}} \mid (\exists^{\geq k} n \leq s)\,(X \in V_n)\} \; .$$

To simplify the calculations, let us identify the sets V_n, W_k, $W_{k,s}$ with their characteristic functions. Thus, $\mu(V_n) = \int_{X \in 2^{\mathbb{N}}} V_n(X)\,dX$, etc. We have

$$\begin{aligned}
2^c \geq \sum_{n=0}^{\infty} \mu(V_n) \quad &\geq \quad \sum_{n=0}^{s} \mu(V_n) \\
&= \quad \sum_{n=0}^{s} \int_X V_n(X)\,dX \\
&= \quad \int_X \sum_{n=0}^{s} V_n(X)\,dX \qquad \text{(the sum is } \geq k \text{ if } X \in W_{k,s}) \\
&\geq \quad \int_X k W_{k,s}(X)\,dX = k\mu(W_{k,s})\,.
\end{aligned}$$

But obviously $W_k = \bigcup_{s=0}^{\infty} W_{k,s}$, hence $\mu(W_k) = \lim_{s\to\infty} \mu(W_{k,s})$, hence our calculation above shows that $2^c \geq k\mu(W_k)$. This proves the claim and completes the proof of Solovay's Lemma. □

Remark 39.7. Solovay's Lemma is a recursion-theoretic refinement of the Borel/Cantelli lemma in probability theory. Solovay's Lemma is frequently used in order to prove that a random X behaves as we would expect.

We now prove another important theorem about Martin-Löf's concept of randomness.

Theorem 39.8. The union of all effectively null sets is effectively null.

Remark 39.9. We can rephrase the theorem as, "there is a *universal* effectively null set." Or, "there is a *universal test for randomness*."

By a *test for randomness* we mean a uniformly Σ^0_1 sequence of sets V_n, $n = 0, 1, 2, \dots$ such that $\mu(V_n) \leq 1/2^n$ for all n. We say that a point $X \in 2^{\mathbb{N}}$ *passes the test* if $X \notin \bigcap_{n=0}^{\infty} V_n$. Our definition of randomness says that X is random if and only if it passes all tests for randomness. The theorem says that there is a particular test for randomness which is *universal* in the following sense: if X passes this particular test for randomness, then it passes all tests for randomness and is therefore random.

Proof. Recall our standard recursive enumeration of all Σ^0_1 subsets of $2^{\mathbb{N}}$,

$$U_e = \{X \in 2^{\mathbb{N}} \mid \varphi_e^{(1),X}(0)\downarrow\}.$$

Define

$$U_{e,s} = \{X \in 2^{\mathbb{N}} \mid \varphi_{e,s}^{(1),X\restriction s}(0)\downarrow\}$$

and note that $U_{e,s}$ is uniformly Σ^0_1 (in fact, uniformly Δ^0_1) and $U_e = \bigcup_{s=0}^{\infty} U_{e,s}$. Furthermore, $\mu(U_{e,s})$ is a rational number and is recursive as a function of e, s. This is because

$$U_{e,s} = \bigcup_{\substack{\varphi_{e,|\sigma|}^{(1),\sigma}(0)\downarrow \\ |\sigma| = s}} N_\sigma$$

and

$$\mu(U_{e,s}) = \sum_{\substack{\varphi_{e,|\sigma|}^{(1),\sigma}(0)\downarrow \\ |\sigma| = s}} \frac{1}{2^s}$$

and the predicates $\varphi_{e,s}^{(1),\sigma}(0)\downarrow$ and $|\sigma| = s$ are recursive. Note that σ ranges over bitstrings of length s.

Given a rational number r, define

$$U_e[r] \;=\; \bigcup_{\mu(U_{e,s}) \leq r} U_{e,s}\;.$$

Intuitively $U_e[r]$ is "U_e enumerated so long as its measure is $\leq r$".

Some easily verified facts are:

1. $U_e[r] \subseteq U_e$.
2. $\mu(U_e[r]) \leq r$.
3. If $\mu(U_e) \leq r$ then $U_e[r] = U_e$.

4. The sets $U_e[r]$ are uniformly Σ^0_1. In other words, the 3-place predicate $P(X, e, r) \equiv X \in U_e[r]$ is Σ^0_1.

Now define

$$V_{e,n} = \begin{cases} U_i[1/2^n] & \text{if } \varphi^{(1)}_e(n) \simeq i \ , \\ \\ \emptyset & \text{if } \varphi^{(1)}_e(n) \uparrow \ . \end{cases}$$

Some easy facts are:

1. $V_{e,n}$ is uniformly Σ^0_1.

   Namely, $X \in V_{e,n} \equiv \exists i \, (\varphi^{(1)}_e(n) \downarrow \simeq i \wedge X \in U_i[1/2^n])$ which is obviously Σ^0_1.

2. $\mu(V_{e,n}) \leq 1/2^n$ for all n.

Therefore, for each e, the sequence of sets $V_{e,n}$, $n = 0, 1, 2, \ldots$ is a test for randomness. Moreover, we claim that all tests for randomness are among these. To see this, suppose that V_n, $n = 0, 1, 2, \ldots$ is a test for randomness, say $V_n = U_{f(n)} = U_{f(n)}[1/2^n]$ for some recursive function $f(n)$. Let e be an index of f, so that $f(n) \simeq \varphi^{(1)}_e(n)$ for all n. Then clearly $V_n = V_{e,n}$ for all n.

Now, to obtain a universal test for randomness, we diagonalize over all tests for randomness by letting

$$\widetilde{V}_n = \bigcup_{e=0}^{\infty} V_{e,e+n+1} \ .$$

Then $X \in \widetilde{V}_n \equiv \exists e \, (X \in V_{e,e+n+1})$ so the sequence $\widetilde{V}_n$, $n = 0, 1, 2, \ldots$, is uniformly Σ^0_1. Moreover

$$\begin{aligned} \mu(\widetilde{V}_n) &\leq \sum_{e=0}^{\infty} \mu(V_{e,e+n+1}) \\ &\leq \sum_{e=0}^{\infty} \frac{1}{2^{e+n+1}} \\ &= \frac{1}{2^n}(\frac{1}{2} + \frac{1}{4} + \cdots) \\ &= \frac{1}{2^n} \end{aligned}$$

so $\widetilde{V}_n$, $n = 0, 1, 2, \ldots$ is a test for randomness. We claim that it is a universal test for randomness. In other words, for all e,

$$\bigcap_{n=0}^{\infty} V_{e,n} \ \subseteq \ \bigcap_{n=0}^{\infty} \widetilde{V}_n \ .$$

This is easily verified: if $X \in \bigcap_{n=0}^{\infty} V_{e,n}$ then $X \in V_{e,e+n+1}$ for all n, hence $X \in \widetilde{V}_n$ for all n, hence $X \in \bigcap_{n=0}^{\infty} \widetilde{V}_n$. This completes the proof. □

Remark 39.10. We view the existence of a universal test for randomness as providing good evidence for the "naturalness" of our concept of randomness. Similarly, Turing's theorem stating the existence of a universal partial recursive function provides good evidence for the "naturalness" of our concept of computable function.

Questions about the "naturalness" of various concepts in mathematics are extremely important. This is because, as mathematicians, we have only a finite amount of time to spend on mathematical research, and therefore it is extremely important to choose the right research topics. If a mathematical concept is "natural" or "interesting", then this suggests that time spent studying the concept will be well spent.

Lecture 26: October 26, 2007

40 Comments on Homework #8

Problem 1

Recall that the machine M is prefix-free if $\mathrm{dom}(M)$ is prefix-free. Define a prefix-free machine, M, by $M(\sigma_1{}^\frown\sigma_2) \simeq U(\sigma_1){}^\frown U(\sigma_2)$. You must check that

1. M is single-valued; i.e., $M(\sigma)$ is well-defined.
2. M is partial recursive.
3. $\mathrm{dom}(M)$ is prefix-free.

Problem 3

Let $r \in \mathbb{R}$ be a real number which is both left r.e. and right r.e. In other words,

$$r = \lim_n a_n, \quad a_n \nearrow \text{ (increasing)}$$

and

$$r = \lim_n b_n, \quad b_n \searrow \text{ (decreasing)}$$

where a_n and b_n are recursive sequences of rational numbers. We need to show that r is a recursive real number.

One solution is to define $f(k) = \mu n\,(|a_n - b_n| < 1/2^k)$. Then f is a recursive function, $r = \lim_{k\to\infty} a_{f(k)}$, and $|a_{f(k)} - r| < 1/2^k$. So the recursive sequence of rational numbers $a_{f(k)}$, $k = 0, 1, 2, \ldots$ witnesses that r is a recursive real number.

Alternatively, we can use the characterization of recursive real numbers as a real number r such that $g(n) =$"nth decimal digit of r" is recursive. It is natural to try something like this:

> "Let $h(k) =$ the least n such that a_n and b_n have the same first $k+1$ digits, and then let $g(k) =$ the kth digit of $a_{h(k)}$."

However, this does not always work! Consider $r = 0.1$ with $a_n = 0.0\underbrace{99\ldots9}_{n}$ and $b_n = 0.1\underbrace{00\ldots0}_{n}1$. In this case and for many other rational r, $h(k)$ is undefined. We can get around this difficulty by considering rational and irrational r as separate cases.

Problem 4

By part (e) we know that for any Π^0_1 set $P \subseteq 2^{\mathbb{N}}$ the real number $\mu(P)$ is right recursively enumerable. Part (f) asks us to find a P such that $\mu(P)$ is not recursive. The easiest example is

$$P \;=\; 2^{\mathbb{N}} \setminus \bigcup_{e \in H} N_{\langle \underbrace{0,\ldots,0}_{e},1\rangle}$$

where H is the Halting Problem. Clearly P is Π^0_1 and

$$\mu(P) \;=\; 1 - \sum_{e \in H} \frac{1}{2^{e+1}}$$

which is a nonrecursive real number.

Problem 5

To find an example of a nonempty Π^0_1 subset of $2^{\mathbb{N}}$ which has no recursive element.

Consider

$$P \;=\; \{X \in 2^{\mathbb{N}} \mid X \text{ separates } A, B\}$$

where A, B is a disjoint pair of recursively inseparable r.e. subsets of $\mathbb{N}$. Obviously P is nonempty has no recursive element. For all $X \in 2^{\mathbb{N}}$ we have

$$X \in P \;\equiv\; \neg\, \exists n\, ((X(n) = 0 \wedge n \in A) \vee (X(n) = 1 \wedge n \in B))\,.$$

Since A and B are Σ^0_1, a Tarski/Kuratowski computation shows that P is Π^0_1.

Alternatively, consider

$$\begin{aligned} Q \;&=\; \{X \in 2^{\mathbb{N}} \mid \forall n\, (X(n) \not\simeq \varphi^{(1)}_n(n))\} \\ &=\; \{X \in 2^{\mathbb{N}} \mid X \text{ is diagonally nonrecursive}\}. \end{aligned}$$

Again, Q is nonempty Π^0_1 and has no recursive element.

Problem 6

Let $P \subseteq \mathbb{N}^{\mathbb{N}}$ be Π^0_2. Say $f \in P \equiv \forall x\, \exists y\, R(f, x, y)$ where R is recursive.

For part (a), let $Q = \{f \oplus g \mid \forall x\,(g(x) = \mu y R(f,x,y))\}$. Clearly Q is Turing isomorphic to P because $g \leq_T f$, hence $f \equiv_T f \oplus g$. To check that Q is Π^0_1, we have $f \oplus g \in Q \equiv \forall x\,(R(f,x,g(x)) \wedge \neg\,\exists y < g(x)\,R(f,x,y))$.

For part (b), let $Q = \{\chi_{G_f} \mid f \in P\}$ where $G_f = \{3^i 5^j \mid f(i) = j\}$ = the "graph" of f. Obviously Q is Turing isomorphic to P, because $f \equiv_T G_f \equiv_T \chi_{G_f}$. It remains to check that Q is Π^0_2. Recall that $f \in P \equiv \forall x\,\exists y\,R(f,x,y)$. Define the partial recursive functional $\Phi(f,x) \simeq \mu y\,R(f,x,y)$. Let e be an index of Φ. By finite approximation we have

$$f \in P \;\equiv\; \forall x\,(\varphi^{(1),f}_e(x)\downarrow) \;\equiv\; \forall x\,\exists n\,(\varphi^{(1),f\restriction n}_{e,n}(x)\downarrow)\;.$$

We know that $X \in Q$ if and only if X is the characteristic function of the "graph" of some function f which belongs to P. Writing this out in detail in terms of e using finite approximation, we have

$$X \in Q \equiv \begin{cases} \forall n\,[(X(n) = 1) \;\Rightarrow\; (n = 3^{(n)_1} \cdot 5^{(n)_2})] \\ \wedge\,\forall m\,\forall n\,[(X(m) = 1 \wedge X(n) = 1 \wedge (m)_1 = (n)_1) \;\Rightarrow\; (m)_2 = (n)_2] \\ \wedge\,\forall i\,\exists j\,\exists n\,[X(n) = 1 \wedge n = 3^i \cdot 5^j] \\ \wedge\,\forall x\,\exists \sigma\,(\forall i < |\sigma|)\,[X(3^i \cdot 5^{\sigma(i)}) = 1 \wedge \varphi^{(1),\sigma}_{e,|\sigma|}(x)\downarrow]\;. \end{cases}$$

These four lines say that X has the form χ_{G_f} where f is single-valued, f is total, and $f \in P$. Thus we see that Q is Π^0_2.

For part (c), the answer is NO! Not every Π^0_2 subset of $2^{\mathbb{N}}$ is Turing isomorphic to a Π^0_1 subset of $2^{\mathbb{N}}$. For an example illustrating this, recall the hierarchy based on the jump operator, $0, 0', 0'', \ldots, 0^{(n)}, \ldots$. Post's Theorem tells us that a set $A \subseteq \mathbb{N}$ is Σ^0_n if and only if $A \leq_m 0^{(n)}$. Define

$$0^{(\omega)} \;=\; \bigoplus_{n=1}^{\infty} 0^{(n)} \;=\; \{3^m \cdot 5^n \mid m \in 0^{(n)}\}\,.$$

Thus $0^{(\omega)}$ is outside the arithmetical hierarchy. Identifying the set $0^{(\omega)}$ with its characteristic function in $2^{\mathbb{N}}$, we can prove that the singleton set $\{0^{(\omega)}\} \subseteq 2^{\mathbb{N}}$ is Π^0_2. On the other hand, we can prove that if $P \subseteq 2^{\mathbb{N}}$ is Π^0_1 and nonempty, then P contains some X which belongs to the arithmetical hierarchy. See also Homework #10, Problems 2 and 3.

41 Homework #10, due November 5, 2007

Exercises 41.1.

1. Let f and g be Turing oracles. Define $f \leq_{LK} g$ to mean that

$$K^g(\tau) \;\leq\; K^f(\tau) + O(1)$$

for all bitstrings τ. Define $f \leq_{LR} g$ to mean that

$$(\forall X \in 2^{\mathbb{N}})\,(\text{if } X \text{ is } g\text{-random then } X \text{ is } f\text{-random}).$$

(a) Show that $f \leq_T g$ implies both $f \leq_{LK} g$ and $f \leq_{LR} g$.

(b) Let $X \in 2^{\mathbb{N}}$ be such that $X \leq_{LK} 0$. Show that X is K-*trivial*, i.e., $K(X \upharpoonright n) \leq K(n) + O(1)$ for all n.

Note: It can be shown that the properties $f \leq_{LK} g$ and $f \leq_{LR} g$ are equivalent to each other. However, they are not equivalent to $f \leq_T g$. In fact, we can find a nonrecursive $X \in 2^{\mathbb{N}}$ such that $X \leq_{LK} 0$. It can be shown that $X \leq_{LK} 0$ if and only if X is K-trivial.

2. For convenience in stating this problem, let us identify subsets of $\mathbb{N}$ with their characteristic functions. In other words, we identify $A \subseteq \mathbb{N}$ with $\chi_A \in 2^{\mathbb{N}}$. Thus $2^{\mathbb{N}}$ is the set of all subsets of $\mathbb{N}$.

   Let $J : 2^{\mathbb{N}} \to 2^{\mathbb{N}}$ be the Turing jump operator:

$$J(X) = X' = H^X = \text{the Halting Problem relative to } X.$$

   Recall that $0^{(1)} = 0' = J(0)$ and in general $0^{(n+1)} = (0^{(n)})' = J(0^{(n)})$ for all n. By Post's Theorem we know that for each $n \geq 1$ the set $0^{(n)}$ is Σ^0_n and not Δ^0_n. Define

$$0^{(\omega)} \;=\; \bigoplus_{n=1}^{\infty} 0^{(n)} \;=\; \{3^m 5^n \mid m \in 0^{(n)}\}\ .$$

   Note that the set $0^{(\omega)}$ is not arithmetical, i.e., it is not Δ^0_n for any n.

   (a) Show that the 2-place predicate $P \subseteq 2^{\mathbb{N}} \times 2^{\mathbb{N}}$ given by

$$P(X, Y) \;\equiv\; J(X) = Y$$

   is Π^0_2.

   (b) Show that for each $n \geq 1$ the singleton set $\{0^{(n)}\}$ is Π^0_2.

   (c) Show that the singleton set $\{0^{(\omega)}\}$ is Π^0_2.

   Note: These singleton sets are subsets of $2^{\mathbb{N}}$.

3. (a) Show that every nonempty Π^0_1 subset of $2^{\mathbb{N}}$ contains a member which is Δ^0_n for some n.

   (b) In part (a), what is the optimal value of n?

   (c) In parts (a) and (b), what if we replace Π^0_1 sets by Π^0_2 sets?

   (d) Is every Π^0_2 subset of $2^{\mathbb{N}}$ Turing isomorphic to a Π^0_1 subset of $2^{\mathbb{N}}$?

4. Let $X \in 2^{\mathbb{N}}$. We say that X is 2-*random* if X is random relative to $0'$. Recall also that X is *weakly* 2-*random* if $X \notin$ any Π^0_2 set of measure 0. Let $\mathbf{a} = \deg_T(X) =$ the Turing degree of X.

(a) Show that if X is 2-random then X is weakly 2-random.

(b) Show that if X is weakly 2-random then $\inf(\mathbf{a}, \mathbf{0}') = 0$.

(c) In part (b) what if we assume only that X is random?

(d) Show that if X is 2-random then $\sup(\mathbf{a}, \mathbf{0}') = \mathbf{a}'$.

(e) In part (d) what if we assume only that X is weakly 2-random?

5. Show that every Π^0_2 subset of $2^{\mathbb{N}}$ includes a $\Sigma^{0,0'}_2$ set of the same measure.

42 A remark on Hilbert's 10th Problem

Professor Kirsten Eisentraeger will talk about Hilbert's 10th Problem at the MASS seminar tomorrow, October 30.

Recall our various characterizations of Σ^0_1 subsets of $\mathbb{N}$. Letting A be a subset of $\mathbb{N}$, we know that

A is Σ^0_1
$\Leftrightarrow$ A is recursively enumerable
$\Leftrightarrow$ A is the range of a partial recursive function
$\Leftrightarrow$ A is the domain of a partial recursive function
$\Leftrightarrow$ A is finite or the range of a 1-1 total recursive function, etc.

Matiyasevich showed:

A is Σ^0_1
$\Leftrightarrow$ A is Diophantine
$\Leftrightarrow$ A is range$(g) \cap \mathbb{N}$ where g is a polynomial with integer coefficients.

Definition 42.1. A k-place number-theoretic predicate $P \subseteq \mathbb{N}^k$ is said to be *Diophantine* if we can find a polynomial equation with $k + n$ variables and integer coefficients such that

$$P(x_1, \ldots, x_k) \equiv \underbrace{\exists y_1 \cdots \exists y_n \, (\underbrace{f(x_1, \ldots, x_k, y_1, \ldots, y_n) = 0}_{\text{Diophantine equation}})}_{\text{Diophantine predicate}}$$

for all $x_1, \ldots, x_k \in \mathbb{N}$.

Obviously, Diophantine predicates are Σ^0_1.

Theorem 42.2 (Matiyasevich's Theorem).

$$P \text{ is } \Sigma^0_1 \quad \text{if and only if} \quad P \text{ is Diophantine.}$$

For example, the Halting Problem $H \subseteq \mathbb{N}$ is Diophantine; hence Hilbert's 10th Problem is unsolvable. In fact, since H is Σ^0_1 complete, we have the following corollary.

Corollary 42.3. Hilbert's 10th Problem $\equiv_m$ the Halting Problem.

43 Initial segment complexity

Let $X \in 2^{\mathbb{N}}$, an infinite sequence of 0's and 1's. We can consider the complexity or prefix-free complexity of the finite initial segments of X: $C(X \upharpoonright n)$ and $K(X \upharpoonright n)$. These quantities are called the *initial segment complexity* of X. The asymptotic behavior of the initial segment complexity of X as n goes to infinity may be viewed as a measure of the "amount of complexity" inherent in X.

Recall that, roughly speaking, $K(|\tau|) \leq K(\tau) \leq |\tau|$ for all strings τ. In other words, $K(n) \leq K(X \upharpoonright n) \leq n$ for $X \in 2^{\mathbb{N}}$ and all positive integers n. And similarly for C. All of these inequalities are modulo an additive constant $O(1)$. Roughly speaking, we have two extreme possibilities for the initial segment complexity of X, given by the following two definitions.

Definition 43.1. We say that X is *K-trivial* if $K(X \upharpoonright n) = K(n) \pm O(1)$ for all n. In other words, $\exists c \, \forall n \, (K(X \upharpoonright n) \leq K(n) + c)$. We define *$C$-trivial* similarly.

Definition 43.2. We say that X is *K-random* if $K(X \upharpoonright n) = n \pm O(1)$ for all n. In other words, $\exists c \, \forall n \, (K(X \upharpoonright n) \geq n - c)$. We define *$C$-random* similarly.

The following facts are known and we shall prove some of them.

1. X is C-trivial $\Leftrightarrow$ X is recursive.
2. $\exists \, K$-trivial X such that X is not recursive.
3. X is K-random $\Leftrightarrow$ X is random.
4. X is C-maximal $\Leftrightarrow$???

In addition to the two extremes of K-triviality and K-randomness, there are many intermediate possibilities. This leads to a fine classification of X in terms of the "amount of complexity" inherent in X, as measured by initial segment complexity.

Lecture 27: October 29, 2007

The following lemma is a useful technical tool in studying prefix-free complexity.

Remark 43.3. Given a prefix-free machine M, we know that $\mathrm{dom}(M)$ is a prefix-free set of bitstrings. Therefore,

$$\sum_{\sigma \in \mathrm{dom}(M)} \frac{1}{2^{|\sigma|}} \;=\; \mu\left(\bigcup_{\sigma \in \mathrm{dom}(M)} N_\sigma \right) \;\leq\; 1\ .$$

This is known as *Kraft's Inequality.* The following lemma is a converse to Kraft's Inequality.

Lemma 43.4 (Kraft/Chaitin Lemma). Let L be a Σ^0_1 subset of $\mathbb{N} \times 2^{<\mathbb{N}}$ such that

$$\sum_{(m,\tau)\in L} \frac{1}{2^m} \ \leq \ 1 \ .$$

Then, we can find a prefix-free machine M such that for each pair $(m, \tau) \in L$ there exists a bitstring σ such that $|\sigma| = m$ and $M(\sigma) \simeq \tau$.

Remark 43.5. Think of L as an abstract specification of a prefix-free machine M. Each pair $(m, \tau) \in L$ specifies that there should exist a bitstring σ of length m such that $M(\sigma) \simeq \tau$. The pairs $(m, \tau) \in L$ are known as the "axioms" of the specification. The conclusion of the lemma says that we can construct a "designer" prefix-free machine M which will satisfy the given specification.

Proof. Since L is recursively enumerable, let (m_i, τ_i), $i = 0, 1, 2, \ldots$ be a recursive enumeration of L. Our assumption on L tells us that

$$\sum_{i=0}^{\infty} \frac{1}{2^{m_i}} \ \leq \ 1 \ .$$

Sublemma. Given a recursive sequence of integers $m_i \geq 0$, $i = 0, 1, 2, \ldots$ such that

$$\sum_{i=0}^{\infty} \frac{1}{2^{m_i}} \ \leq \ 1 \ ,$$

we can effectively find a recursive, prefix-free sequence of bitstrings σ_i, $i = 0, 1, 2, \ldots$ such that $|\sigma_i| = m_i$ for all i.

Once we have this, we can simply define $M(\sigma_i) \simeq \tau_i$ for all i. It is then obvious that M is a prefix-free machine as desired.

Proof of sublemma. We define σ_i, $i = 0, 1, 2, \ldots$ by recursion on i. In defining σ_k, we may use course-of-values recursion and assume that we already know σ_i, $0 \leq i < k$. We may also assume inductively that we have another finite, prefix-free set of bitstrings D_k with the following properties:

1. $D_k \cap \{\sigma_i \mid 0 \leq i < k\} = \emptyset$.
2. $D_k \cup \{\sigma_i \mid 0 \leq i < k\}$ is a partition.
3. All of the bitstrings in D_k have different lengths.

Definition 43.6. A *partition* is a finite, prefix-free set of bitstrings, call it F, such that

$$2^{\mathbb{N}} = \bigcup_{\sigma\in F} N_\sigma \ .$$

To be finished next time . . . . □

□

Lecture 28: October 31, 2007

We resume our analysis of prefix-free complexity.

Remark 43.7. Given a prefix-free sequence of bitstrings σ_i, $i = 0, 1, 2, \ldots$, we have

$$\sum_{i=0}^{\infty} \frac{1}{2^{|\sigma_i|}} \ \leq \ 1$$

because the sum on the left is just $\mu(\bigcup_{i=0}^{\infty} N_{\sigma_i})$ which is ≤ 1 because $\bigcup_{n=0}^{\infty} N_{\sigma_i} \subseteq 2^{\mathbb{N}}$ and $\mu(2^{\mathbb{N}}) = 1$. This is the Kraft Inequality. The following lemma is a converse of this remark.

Lemma 43.8. Given a sequence of positive integers m_i, $i = 0, 1, 2, \ldots$ such that

$$\sum_{i=0}^{\infty} \frac{1}{2^{m_i}} \ \leq \ 1 \ ,$$

we can find a prefix-free sequence of bitstrings σ_i, $i = 0, 1, 2, \ldots$ such that $|\sigma_i| = m_i$ for all i. Moreover, if the sequence m_i, $i = 0, 1, 2, \ldots$ is recursive, we can take the sequence σ_i, $i = 0, 1, 2, \ldots$ to be recursive.

Proof. The proof is based on the following definition.

Definition 43.9. A *partition* is a finite, prefix-free set of bitstrings, F, such that

$$2^{\mathbb{N}} \ = \ \bigcup_{\sigma \in F} N_\sigma \, .$$

We will construct σ_i, $i = 0, 1, 2, \ldots$ by induction. At each stage k, in defining σ_k we may assume that the prefix-free finite sequence of bitstrings σ_i, $i < k$, is already known. In addition we assume that we have a finite prefix-free set of bitstrings D_k with the following properties:

1. $D_k \cap \{\sigma_i \mid i < k\} = \emptyset$.

2. $D_k \cup \{\sigma_i \mid i < k\}$ is a partition.

3. All of the strings in D_k are of different lengths.

Example 43.10. To illustrate the construction, consider $m_0 = 2, m_1 = 4, m_2 = 3, m_3 = 2, \ldots$ such that

$$\sum_{i=0}^{\infty} \frac{1}{2^{m_i}} \ = \ \frac{1}{4} + \frac{1}{16} + \frac{1}{8} + \frac{1}{4} + \cdots \ \leq \ 1 \, .$$

In constructing the sequence of bitstrings $\sigma_0, \sigma_1, \sigma_2, \sigma_3, \ldots$, the idea is to always choose the leftmost available branch on the binary tree at each step (i.e., the branch with the most 0's). The sets D_k are constructed by taking the shortest possible branches from the right side of the binary tree until D_k satisfies the induction hypothesis.

Given $m_0 = 2$, pick $\sigma_0 = \langle 0, 0\rangle$ and $D_1 = \{\langle 1\rangle, \langle 0, 1\rangle\}$.
Then, given $m_1 = 4$, pick $\sigma_1 = \langle 0, 1, 0, 0\rangle$ and $D_2 = \{\langle 1\rangle, \langle 0, 1, 1\rangle, \langle 0, 1, 0, 1\rangle\}$.
Then, given $m_2 = 3$, pick $\sigma_2 = \langle 0, 1, 1\rangle$ and $D_3 = \{\langle 1\rangle, \langle 0, 1, 0, 1\rangle\}$.
Then, given $m_3 = 2$, pick $\sigma_3 = \langle 1, 0\rangle$ and $D_4 = \{\langle 1, 1\rangle, \langle 0, 1, 0, 1\rangle\}$.
Etc.

Formally, at stage $k+1$, suppose we already have σ_i, $i < k$ and D_k as above.

We claim that $m_k \geq \min\{|\rho| \mid \rho \in D_k\}$. Otherwise $m_k < \min\{|\rho| \mid \rho \in D_k\}$, hence

$$\frac{1}{2^{m_k}} > \sum_{\rho \in D_k} \frac{1}{2^{|\rho|}}$$

in view of the requirement that all strings in D_k are of different lengths. Hence

$$\sum_{i=0}^{k} \frac{1}{2^{m_i}} \;=\; \frac{1}{2^{m_k}} + \sum_{i=0}^{k-1} \frac{1}{2^{m_i}} \;>\; \sum_{\rho \in D_k} \frac{1}{2^{|\rho|}} + \sum_{i=0}^{k-1} \frac{1}{2^{|\sigma_i|}} \;=\; 1$$

a contradiction. This proves the claim.

By the claim, let $\rho_k \in D_k$ be of maximal length such that $|\rho_k| \leq m_k$. Letting

$$\sigma_k = \rho_k{}^\frown\langle\underbrace{0, \ldots, 0}_{m_k - |\rho_k|}\rangle$$

we see that $|\sigma_k| = m_k$ and $\sigma_k \mid \sigma_i$ for all $i < k$. Letting

$$D_{k+1} = D_k \setminus \{\rho_k\} \cup \{\rho_k{}^\frown\langle\underbrace{0, \ldots, 0}_{j}, 1\rangle \mid j < m_k - |\rho_k|\}$$

it is straightforward to check that properties 1,2, and 3 hold with D_{k+1} in place of D_k. This completes the proof. □

Theorem 43.11 (Kraft/Chaitin Theorem). Given a Σ^0_1 set $L \subseteq \mathbb{N} \times 2^{<\mathbb{N}}$ such that

$$\sum_{(m,\tau) \in L} \frac{1}{2^m} \;\leq\; 1\;,$$

we can find a prefix-free machine M such that for all $(m, \tau) \in L$ there exists σ such that $|\sigma| = m$ and $M(\sigma) \simeq \tau$.

Proof. Since L is recursively enumerable, let (m_i, τ_i), $i = 0, 1, 2, \ldots$ be a one-to-one recursive enumeration of L. Apply the previous lemma to find a recursive, prefix-free sequence of bitstrings σ_i, $i = 0, 1, 2, \ldots$ such that $|\sigma_i| = m_i$ for all i. Let $M(\sigma_i) \simeq \tau_i$ for all i. □

Remark 43.12. The idea of the Kraft/Chaitin Theorem is that L is an abstract specification of a prefix-free machine. The pairs $(m, \tau) \in L$ are called the "axioms" of the specification. Each axiom (m, τ) says that our prefix-free machine M is intended to have $M(\sigma) \simeq \tau$ for some σ of length m. The theorem asserts that we can find a prefix-free machine which meets all of these requirements.

The Kraft/Chaitin Theorem has the following consequence for prefix-free complexity, K.

Corollary 43.13. Let L be a Σ^0_1 subset of $\mathbb{N} \times 2^{<\mathbb{N}}$ such that

$$\sum_{(m,\tau)} \frac{1}{2^m} \ < \ \infty \ .$$

Then for all $(m,\tau) \in L$ we have $K(\tau) \leq m + O(1)$.

Proof. Let c be such that

$$\sum_{(m,\tau)\in L} \frac{1}{2^m} \ \leq \ 2^c \ < \ \infty \ .$$

Then

$$\sum_{(m,\tau)\in L} \frac{1}{2^{m+c}} \ \leq \ 1$$

so by the Kraft/Chaitin Theorem, let M be a prefix-free machine such that for all $(m,\tau) \in L$ there exists σ such that $|\sigma| = m + c$ and $M(\sigma) \simeq \tau$. Then for all $(m,\tau) \in L$ we have $K(\tau) \leq m + c + O(1) = m + O(1)$. □

Lecture 29: November 2, 2007

We now exhibit a close connection between randomness and Kolmogorov complexity. Recall that $X \in 2^{\mathbb{N}}$ is said to be *K-random* if $K(X \upharpoonright n) \geq n - O(1)$. In other words, $\exists c \forall n \, (K(X \upharpoonright n) \geq n - c)$.

Theorem 43.14 (Schnorr's Theorem). X is random $\Leftrightarrow$ X is K-random.

Proof. ($\Rightarrow$) Assume X is random. Let $V_c = \{X \in 2^{\mathbb{N}} \mid \exists n \, (K(X \upharpoonright n) < n - c)\}$. Note V_c is uniformly Σ^0_1 for all $c = 0, 1, 2, \ldots$, namely

$$\begin{aligned} X \in V_c \ &\equiv \ \exists \tau \, (\tau \subset X \wedge K(\tau) < |\tau| - c) \\ &\equiv \ \exists \tau \, (\tau \subset X \wedge \exists \sigma \, (|\sigma| < |\tau| - c \wedge U(\sigma) \simeq \tau)) \\ &\equiv \ \Sigma^0_1 \end{aligned}$$

where U is a universal prefix-free machine.

We claim that $\mu(V_c) < 1/2^c$. To see this, for each τ such that $K(\tau) < |\tau| - c$ choose a σ such that $U(\sigma) \simeq \tau$ and $|\sigma| < |\tau| - c$. We then have

$$\sum_{\sigma} \frac{1}{2^{|\sigma|}} \ \leq \ 1$$

by the Kraft inequality. It follows that

$$\mu(V_c) \leq \sum_{\tau} \frac{1}{2^{|\tau|}} \ < \ \sum_{\sigma} \frac{1}{2^{|\sigma|+c}} \ \leq \ \frac{1}{2^c}$$

which proves the claim.

Thus, X random implies $X \notin V_c$ for some c, which means that $\forall n\, K(X \restriction n) \geq n - c$. Hence $K(X \restriction n) \geq n - O(1)$, i.e., X is K-random, Q.E.D.

($\Leftarrow$) Assume X is not random. Then $X \in \bigcap_{n=0}^{\infty} V_n$ where V_n is uniformly Σ^0_1 and $\mu(V_n) \leq 1/2^n$. Let T_n, $n = 0, 1, 2, \ldots$ be a uniformly recursive, prefix-free set of bitstrings such that $V_n = \bigcup_{\tau \in T_n} N_\tau$. We have

$$\sum_{n=0}^{\infty} \sum_{\tau \in T_{2n}} \frac{1}{2^{|\tau|-n}} \;=\; \sum_{n=0}^{\infty} 2^n \mu(V_{2n}) \;\leq\; \sum_{n=0}^{\infty} 2^n \left(\frac{1}{2^{2n}}\right) \;=\; \sum_{n=0}^{\infty} \frac{1}{2^n} \;=\; 2\,.$$

It follows by Kraft/Chaitin (see Corollary 43.13) that $K(\tau) \leq |\tau| - n + O(1)$ for all such pairs (τ, n). That is, for some fixed c, $K(\tau) \leq |\tau| - n + c$ for all n and all $\tau \in T_{2n}$. But then, since $X \in V_{2n}$ for all n, we have $\forall n\, \exists m\, (K(X \restriction m) \leq m - n + c)$. In other words, it is not the case that $K(X \restriction m) > m - c - O(1)$ for all m. Thus X is not K-random, Q.E.D. □

Remark 43.15. Schnorr's Theorem exhibits a close relationship between our two approaches to randomness. We may view prefix-free complexity $K(\tau)$ as the length of the smallest compressed version of τ. So randomness can be seen not only in terms of probability, but also in terms of compressibility of strings.

Remark 43.16. As usual, we can relativize all of these concepts and theorems to an arbitrary oracle $f \in \mathbb{N}^{\mathbb{N}}$. This goes as follows.

Definition 43.17. A sequence of sets $V_n \subseteq 2^{\mathbb{N}}$, $n = 0, 1, 2, \ldots$ is said to be *uniformly* $\Sigma^{0,f}_1$ if .... The final problem in Homework #9 was to show that three reasonable definitions of this concept coincide.

Definition 43.18. We say that X is *f-random* (i.e., *random relative to f*) if there is no uniformly $\Sigma^{0,f}_1$ sequence of sets V_n, $n = 0, 1, 2, \ldots$ such that $X \in \bigcap_{n=0}^{\infty} V_n$ and $\mu(V_n) \leq 1/2^n$ for all n.

Definition 43.19. An *f-machine* is a partial f-recursive function from bitstrings to bitstrings, $M :\subseteq 2^{<\mathbb{N}} \to 2^{<\mathbb{N}}$.

Definition 43.20. M is *prefix-free* if ....

Definition 43.21. A *universal prefix-free f-machine* is ....

Definition 43.22. We define $K^f(\tau) = \min\{|\sigma| | U^f(\sigma) \simeq \tau\}$ where U^f is a universal prefix-free f-machine.

Theorem 43.23 (relativization of Schnorr's Theorem). X is f-random $\Leftrightarrow$ X is K^f-random.

44 Solutions for Homework #9

Problem 1

For all $n \geq 1$ and all rational $\epsilon > 0$, let $V_{n,\epsilon}$ be the set of $X \in 2^{\mathbb{N}}$ such that

$$\left| \frac{\sum_{i=0}^{n-1} X(i)}{n} - \frac{1}{2} \right| \; > \; \epsilon \; .$$

Note that $V_{n,\epsilon}$ is uniformly Σ^0_1. Hoeffding's Inequality says that

$$\mu(V_{n,\epsilon}) \; \leq \; \frac{2}{e^{2n\epsilon^2}} \; = \; 2\left(\frac{1}{e^{2\epsilon^2}}\right)^n \; .$$

Since $1/e^{2\epsilon^2}$ is < 1, we have

$$\sum_{n=0}^{\infty} \left(\frac{1}{e^{2\epsilon^2}}\right)^n \; < \; \infty \qquad \text{(geometric series)}$$

hence $\sum_{n=0}^{\infty} \mu(V_{n,\epsilon}) < \infty$. Thus, by Solovay's Lemma, if X is random then $X \in V_{n,\epsilon}$ for only finitely many n. Since this holds for all $\epsilon > 0$, we see that X satisfies the Strong Law of Large Numbers,

$$\lim_{n\to\infty} \frac{\sum_{i=0}^{n-1} X(i)}{n} \; = \; \frac{1}{2} \; .$$

Problem 3

We modify the argument for Problem 1 by letting ϵ depend on n. For instance, we can define

$$\epsilon_n \; = \; \sqrt{\frac{\log n}{n}} \; .$$

Letting $V_n = V_{n,\epsilon_n}$ we see that

$$\sum_{n=0}^{\infty} \mu(V_n) \; \leq \; \sum_{n=0}^{\infty} \frac{2}{e^{2n\epsilon_n{}^2}} \; = \; \sum_{n=0}^{\infty} \frac{2}{e^{2\log n}} \; = \; \sum_{n=0}^{\infty} \frac{2}{n^2} \; < \; \infty$$

(p-series, $p = 2$). So again $X \in V_n$ for only finitely many n. In other words,

$$\left| \frac{\sum_{i=0}^{n-1} X(i)}{n} - \frac{1}{2} \right| \; \leq \; \sqrt{\frac{\log n}{n}}$$

for all sufficiently large n.

Problem 5

For all $X \in 2^{\mathbb{N}}$ we have $X = X_0 \oplus X_1$ where $X_0(n) = X(2n)$ and $X_1(n) = X(2n+1)$. Thus X_0 and X_1 are the even and odd parts of X, respectively.

Problem 4 was to show that if X is random then X_0 and X_1 are Turing incomparable.

Problem 5 was to find a weakly random X such that $X_0 \equiv_T X_1$. We construct X by finite approximation.

Stage 0. Let $\sigma_0 = \langle\rangle$.

Stage $e+1$. Assume we already know σ_e.

Case 1: $\exists \sigma \supseteq \sigma_e{}^\frown\langle 1,1\rangle$ such that $\varphi^{(1),\sigma}_{e,|\sigma|}(0)\downarrow$. Let σ_{e+1} = the least such σ which is of even length.

Case 2: not Case 1. Let $\sigma_{e+1} = \sigma_e{}^\frown\langle 0,0\rangle$.

Finally let $X = \bigcup_{e=0}^{\infty} \sigma_e$. Note that $|\sigma_e|$ is even for all e.

Recall our standard recursive enumeration of all Π^0_1 subsets of $2^{\mathbb{N}}$, namely $P_e = \{X \in 2^{\mathbb{N}} \mid \varphi^{(1),X}_e(0)\uparrow\}$. Consider what happened at stage $e+1$. If Case 1 holds, then $X \notin P_e$. If Case 2 holds, then $N_{\sigma_e{}^\frown\langle 1,1\rangle} \subseteq P_e$, hence $\mu(P_e) > 0$. Thus X is weakly random.

To see that $X_0 \equiv_T X_1$, it suffices to show that the entire construction σ_e, $e = 0, 1, 2, \ldots$ is both $\leq_T X_0$ and $\leq_T X_1$. Assume that we are using one of the oracles X_0 or X_1 and we have already computed σ_e. Note that Case 1 holds at stage $e+1$ if and only if $X_0(|\sigma_e|/2) = 1$, if and only if $X_1(|\sigma_e|/2) = 1$. Thus we can use our oracle to tell which case we are in. If we are in Case 1, we can recursively search for σ_{e+1} = the least $\sigma \supseteq \sigma_e{}^\frown\langle 1,1\rangle$ of even length such that $\varphi^{(1),\sigma}_{e,|\sigma|}(0)\downarrow$. Otherwise we are in Case 2 and $\sigma_{e+1} = \sigma_e{}^\frown\langle 0,0\rangle$. Either way we have now computed σ_{e+1}.

Problem 6

Let X be weakly random. Suppose W_e is an infinite r.e. set. Since W_e is infinite,

$$P \;=\; \{X \in 2^{\mathbb{N}} \mid \forall n\,(n \in W_e \Rightarrow X(n) = 1)\}$$

is of measure 0. Since W_e is Σ^0_1, a Tarski/Kuratowski computation shows that P is Π^0_1. Hence $X \notin P$, hence $X(n) = 0$ for at least one $n \in W_e$. Similarly we can show that $X(n) = 1$ for at least one $n \in W_e$. Thus X is the characteristic function of a biimune set.

45 Homework #11, due November 12, 2007

Exercises 45.1. For $f, g \in \mathbb{N}^{\mathbb{N}}$ say that f is *majorized* by g if $f(n) < g(n)$ for all n.

1. If $P(f, g, -)$ is a Π^0_1 predicate, prove that the predicate

$$Q(g, -) \;\equiv\; \exists f\,(P(f, g, -) \wedge f \text{ is majorized by } g)$$

is again Π^0_1.

Note: This is a generalization of the Magic Lemma, Lemma 48.3 in the Lecture Notes. You can prove it by imitating the the proof of Lemma 48.3.

2. (a) Show that the result of Problem 1 holds if we replace Π^0_1 by Σ^0_2.

   (b) Show that the result does not hold if we replace Π^0_1 by Π^0_2.

   In fact, we can find a Π^0_2 predicate $P(X, -)$ with X ranging over $2^{\mathbb{N}}$ such that the predicate $\exists X\, P(X, -)$ is not arithmetical, i.e., it is not Π^0_n or Σ^0_n for any n.

3. Let $P \subseteq 2^{\mathbb{N}}$ be Π^0_1. Let $\Phi(X, n)$ be a partial recursive functional such that $\Phi(X, n)\downarrow$ for all $X \in P$ and all n. Find a total recursive function $g(n)$ which majorizes $\Phi(X, n)$ for all $X \in P$ and all n.

4. An oracle X is said to be *hyperimmune-free* (sorry for the awkward terminology) if each $f \leq_T X$ is majorized by some recursive function.

   Note: This is another example of a "lowness property" of X.

   (a) Let $P \subseteq 2^{\mathbb{N}}$ be nonempty and Π^0_1. Prove that there exists $X \in P$ such that X is hyperimmune-free. This result is known as the Hyperimmune-Free Basis Theorem.

   Hint: Use Π^0_1 approximation as in the Low Basis Theorem.

   (b) Deduce that we can find a random X which is hyperimmune-free.

5. Prove that if $0 <_T X \leq_T 0'$ then X is not hyperimmune-free.

   Note: This prevents us from combining the Low Basis Theorem and the Hyperimmune-Free Basis Theorem into one theorem.

   Hint for the proof: By Post's Theorem X is Δ^0_2. Deduce that the singleton set $\{X\}$ is Π^0_2. Use this to find $f \equiv_T X$ such that the singleton set $\{f\}$ is Π^0_1. If such an f is majorized by a recursive function, use the result of Problem 1 to show that f is recursive.

6. (Extra Credit)

   (a) Prove that if X is 2-random then X is not hyperimmune-free.

   (b) What if we assume only that X is weakly 2-random?

7. (a) Prove that if Y is nonrecursive then $\mu(\{X \in 2^{\mathbb{N}} \mid Y \not\leq_T X\}) = 1$.

   (b) Deduce that for each nonrecursive Y we can find a random X such that $Y \not\leq_T X$.

   (c) More generally, prove the following. Given a sequence of nonrecursive oracles Y_i, $i = 0, 1, 2, \ldots$, we can find an X which is n-random for all n and such that $Y_i \not\leq_T X$ for all i.

Note: It can be shown that for all Y we can find a random X such that $Y \leq_T X$. In fact, each Turing degree $\geq \mathbf{0}'$ contains a random X. However, this does not hold for weakly 2-random X's, because all such X's are Turing incomparable with $0'$.

8. (a) Assume that $P \subseteq 2^{\mathbb{N}}$ is Π^0_1 and

$$\neg \exists X \, (X \in P \wedge X \text{ is recursive}).$$

Find a nonrecursive Y such that

$$\neg \exists X \, (X \in P \wedge X \leq_T Y).$$

Hint: Use finite approximation.

(b) Find a nonrecursive Y such that

$$\neg \exists X \, (X \text{ is random} \wedge X \leq_T Y).$$

Hint: Use the fact that $\{X \mid X \text{ is random}\}$ is the union of a sequence of Π^0_1 sets.

Lecture 30: November 5, 2007

46 Turing degrees of random sequences

Let X be an infinite sequence of 0's and 1's which is random in the sense of Martin-Löf. What can we say about the Turing degree of X? We shall obtain some answers to this question.

Remark 46.1. We already know that if X is random then X is nonrecursive, i.e., the Turing degree of X is nonzero. We shall prove:

1. There exist random X's whose Turing degree is $< \mathbf{0}'$.
2. There exist random X's whose Turing degree is incomparable with $\mathbf{0}'$ (see below for more details).
3. There exist nonzero Turing degrees $\mathbf{b}$ such that no Turing degree $\leq \mathbf{b}$ contains a random X.
4. Every Turing degree $\geq \mathbf{0}'$ contains a random X (if time permits).

Definition 46.2. A Turing degree $\mathbf{a}$ is said to be *low* if $\mathbf{a}' = \mathbf{0}'$.

Definition 46.3. For $f, g \in \mathbb{N}^{\mathbb{N}}$ we say f *is dominated by* g if $f(n) < g(n)$ for all sufficiently large n. We say that f *is majorized by* g if $f(n) < g(n)$ for all n.

Definition 46.4. A Turing degree $\mathbf{a}$ is said to be *hyperimmune-free* if every function of degree $\mathbf{a}$ is dominated by a recursive function. Equivalently, every function of degree $\mathbf{a}$ is majorized by a recursive function.

Remark 46.5. Both of these properties, lowness and hyperimmune-freeness, say that the Turing degree $\mathbf{a}$ is in some sense close to $\mathbf{0}$. We shall see:

1. There exist random X's which are low.
2. There exist random X's which are hyperimmune-free.
3. There do not exist random X's which are both low and hyperimmune-free.

47 Compactness of $2^{\mathbb{N}}$

In order to study the Turing degrees of random sequences, it is convenient to use the fact that the Cantor space $2^{\mathbb{N}}$ is compact. This is embodied in the following theorem.

Theorem 47.1 (compactness of $2^{\mathbb{N}}$). If S is a set of bitstrings such that $2^{\mathbb{N}} = \bigcup_{\sigma \in S} N_\sigma$, then $2^{\mathbb{N}} = \bigcup_{\sigma \in F} N_\sigma$ for some finite $F \subseteq S$. In other words,

> "Every covering of $2^{\mathbb{N}}$ by neighborhoods contains a finite subcovering."

Note that this does not hold for the Baire space, $\mathbb{N}^{\mathbb{N}}$. For example, $\mathbb{N}^{\mathbb{N}} = \bigcup_{i=0}^{\infty} N_{\langle i \rangle}$ but obviously there is no finite subcovering.

In order to prove Theorem 47.1, we shall first prove a combinatorial lemma concerning trees. Recall that a *tree* is a set $T \subseteq \mathbb{N}^{<\mathbb{N}}$ which is closed under initial segments, i.e., $\sigma \subset \tau, \tau \in T \Rightarrow \sigma \in T$.

Definition 47.2. A tree T is said to be *finitely branching* if for every $\sigma \in T$ there are only finitely many n such that $\sigma^\frown\langle n \rangle \in T$.

Define an *immediate extension* of a string σ to be any string of the form $\sigma^\frown\langle n \rangle$ for some n. Note that a tree T is finitely branching if and only if each string in T has only finitely many immediate extensions in T.

Example 47.3. The full binary tree $2^{<\mathbb{N}}$ is finitely branching, because for any bitstring σ the only immediate extensions of σ which are bitstrings are $\sigma^\frown\langle 0 \rangle$ and $\sigma^\frown\langle 1 \rangle$.

Note also that any subtree of a finitely branching tree is finitely branching. In particular, any subtree of the full binary tree is finitely branching.

Lemma 47.4 (König's Lemma). Let T be a finitely branching tree. Then T is infinite $\Leftrightarrow$ T has a path.

Note that König's Lemma fails badly for trees that are not finitely branching. For example, the tree $\{\langle\rangle, \langle 0 \rangle, \langle 1 \rangle, \langle 2 \rangle, \ldots\}$ is infinite and not finitely branching and has no path, in fact it contains no string of length 2.

Proof of König's Lemma. The $\Leftarrow$ is obvious. If f is a path through T, then T contains the strings $f \restriction n$, $n = 0, 1, 2, \ldots$, hence T is infinite.

To prove ⇒, let T be an infinite, finitely branching tree. Define

$$\widehat{T} = \{\sigma \in T \mid T \text{ contains infinitely many extensions of } \sigma\}\,.$$

Note that $\widehat{T}$ is a subtree of T.

We claim that the empty string $\langle\rangle$ belongs to $\widehat{T}$. This is obvious, because T is infinite.

We claim that any $\sigma \in \widehat{T}$ has an immediate extension belonging to $\widehat{T}$. This is because any extension of σ in T is an extension of some immediate extension of σ in T. Since σ has infinitely many extensions in T, and σ has only finitely many immediate extensions in T (because T is finitely branching), it follows by the Pigeonhole Principle that at least one of the immediate extensions of σ has infinitely many extensions in T, i.e., it belongs to $\widehat{T}$.

Now, to construct a path through T, it suffices to construct a path through $\widehat{T}$. Start with $\sigma_0 = \langle\rangle \in \widehat{T}$. Given $\sigma_i \in \widehat{T}$, let $n_i =$ some n such that $\sigma_i{}^\frown\langle n\rangle \in \widehat{T}$, and let $\sigma_{i+1} = \sigma_i{}^\frown\langle n_i\rangle$. Thus $\sigma_0, \sigma_1, \sigma_2, \ldots$ is an infinite path through $\widehat{T}$. Actually, the infinite path f is defined by $f(i) = n_i$ for all i. □

Proof of Theorem 47.1. Given a covering $2^{\mathbb{N}} = \bigcup_{\sigma \in S} N_\sigma$ where S is a set of bitstrings, let

$$T \ = \ \{\tau \in 2^{<\mathbb{N}} \mid \neg\,(\exists m \leq |\tau|)\,(\tau \restriction m \in S)\}\ .$$

Clearly T is a tree. T is finitely branching, because $T \subseteq 2^{<\mathbb{N}}$. T has no path, because if $X \in 2^{\mathbb{N}}$ were a path through T we would have $X \restriction n \in T$ for all n, hence $X \restriction n \notin S$ for all n, hence $X \notin N_\sigma$ for all $\sigma \in S$, a contradiction.

Therefore, by König's Lemma, T is finite. Let n be such that T contains no bitstring of length n. In other words, every bitstring of length n has an initial segment belonging to S. Hence,

$$2^{\mathbb{N}} \ = \ \bigcup_{\sigma \in S, |\sigma| \leq n} N_\sigma \ = \ \bigcup_{\sigma \in F} N_\sigma$$

where $F = \{\sigma \in S \mid |\sigma| \leq n\}$. Note that F is finite of cardinality $\leq 2^n$. □

Corollary 47.5. If $2^{\mathbb{N}} = \bigcup_{n=0}^{\infty} V_n$ with V_n open, then $2^{\mathbb{N}} = \bigcup_{n=0}^{k} V_n$ for some k. In other words, $2^{\mathbb{N}}$ is *compact* in the usual topological sense:

> "Every open covering of $2^{\mathbb{N}}$ has a finite subcovering."

Proof. This follows easily from Theorem 47.1, because an open set is the union of a sequence of neighborhoods. □

Lecture 31: November 7, 2007

Remark 47.6. Here are a few announcements about upcoming seminars and colloquia relevant to this course.

- Tomorrow, MASS Colloquium: Professor Alexandra Shlapentokh, expert on Hilbert's 10th Problem.
- Tuesday, MASS Seminar: Professor Peter Cholak, expert on randomness, Kolmogorov complexity, etc.
- Tuesday, Logic Seminar: Professor Peter Cholak (2:30-3:45 in 106 McAllister).

We now continue with our discussion of the fact that $2^{\mathbb{N}}$ is compact.

We have proved:

Corollary 47.7. If $2^{\mathbb{N}} = \bigcup_{n=0}^{\infty} V_n$ where each V_n is an open subset of $2^{\mathbb{N}}$, then $2^{\mathbb{N}} = \bigcup_{n=0}^{k} V_n$ for some k. In other words:

> "Every open covering of $2^{\mathbb{N}}$ has a finite subcovering."

Since a closed set is the complement of an open set, we can restate the previous corollary in terms of closed sets, as follows:

Corollary 47.8. Assume that

$$Q_0 \supseteq Q_1 \supseteq \cdots \supseteq Q_n \supseteq Q_{n+1} \supseteq \cdots$$

is a descending sequence of closed sets in $2^{\mathbb{N}}$ and $Q_n \neq \emptyset$ for all n. Then $\bigcap_{n=0}^{\infty} Q_n \neq \emptyset$.

Proof. Look at the open sets $V_n = 2^{\mathbb{N}} \setminus Q_n$ and apply the previous corollary. □

Remark 47.9. Recall also that Σ_1^0 sets are open, and Π_1^0 sets are closed. Hence, the above corollaries apply to these sets as well.

48 Σ_1^0 and Π_1^0 predicates in $2^{\mathbb{N}}$

We now use the compactness of $2^{\mathbb{N}}$ to draw some interesting consequences concerning Σ_1^0 and Π_1^0 predicates.

Recall that we are dealing with three important spaces: $\mathbb{N}^{\mathbb{N}}$ (the Baire space), $2^{\mathbb{N}}$ (the Cantor space), and $\mathbb{N}$ (the natural numbers). Actually we are dealing with "mixed" predicates $S \subseteq (\mathbb{N}^{\mathbb{N}})^m \times (2^{\mathbb{N}})^l \times \mathbb{N}^k$. For convenience in stating the following lemma, let us abbreviate $S(f_1, \ldots, f_m, X_1, \ldots, X_l, n_1, \ldots, n_k)$ as $S(-,-,-)$.

Lemma 48.1. Let f be a variable ranging over $\mathbb{N}^{\mathbb{N}}$. Then any Σ_1^0 predicate $S(f,-,-,-)$ can be written in the form

$$S(f,-,-,-) \;\equiv\; \exists n \, R(f \restriction n, -,-,-)$$

where $R(\sigma,-,-,-)$ is a recursive predicate and σ is a variable ranging over strings.

Proof. This follows easily from the idea of finite approximation. In detail we have

$$\begin{aligned} S(f,-,-,-) &\equiv \varphi_e^{(k),f\oplus-\oplus-}(-)\downarrow \\ &\equiv \exists s\ \varphi_{e,s}^{(k),f\restriction s\oplus-\restriction s\oplus-\restriction s}(-)\downarrow \\ &\equiv \exists s\ R(f\restriction s,-,-,-) \end{aligned}$$

where $R(\sigma,-,-,-)$ is the recursive predicate $\varphi_{e,|\sigma|}^{(k),\sigma\oplus-\restriction|\sigma|\oplus-\restriction|\sigma|}(-)\downarrow$. □

Lemma 48.2 (boundedness principle). Let X be a variable ranging over $2^{\mathbb{N}}$. Let $S(X,n,-)$ be a Σ^0_1 predicate. Then

$$\forall X\,\exists n\,S(X,n,-) \;\equiv\; \exists k\,\forall X\,(\exists n<k)\,S(X,n,-)\ .$$

In other words, the n's on the left hand side are bounded.

Proof. Fix $-$ and let $V_n=\{X\in 2^{\mathbb{N}}\mid S(X,n,-)\}$. Since S is Σ^0_1, V_n is open. By Lemma 48.1 we have

$$\begin{aligned} \forall X\,\exists n\,S(X,n,-) &\equiv \forall X\,\exists n\,(X\in V_n) \\ &\equiv 2^{\mathbb{N}}=\textstyle\bigcup_{n=0}^{\infty}V_n \\ &\equiv \exists k\ \ 2^{\mathbb{N}}=\textstyle\bigcup_{n=0}^{k-1}V_n \qquad \text{(by compactness)} \\ &\equiv \exists k\,\forall X\,(\exists n<k)\ S(X,n,-)\,. \end{aligned}$$

□

Lemma 48.3 (the Magic Lemma). Let X be a variable ranging over $2^{\mathbb{N}}$.

1. If $S(X,-)$ is Σ^0_1 then $\forall X\,S(X,-)$ is Σ^0_1.
2. If $P(X,-)$ is Π^0_1 then $\exists X\,P(X,-)$ is Π^0_1.

In other words,

1. The class of Σ^0_1 predicates is closed under $\forall X$.
2. The class of Π^0_1 predicates is closed under $\exists X$.

These closure properties are useful in Tarski/Kuratowski computations.

Proof. We prove only part 1. Part 2 follows by duality. Let $S(X,-)$ be a Σ^0_1 predicate. Use Lemma 48.1 to write

$$S(X,-) \;\equiv\; \exists n\,R(X\restriction n,-)$$

where $R(\sigma, -)$ is a recursive predicate. By Lemma 48.2 we have

$$\begin{aligned}
\forall X\, S(X,-) &\equiv \forall X\, \exists n\, R(X \upharpoonright n, -) \\
&\equiv \exists k\, \forall X\, (\exists n < k)\, R(X \upharpoonright n, -) \\
&\equiv \exists k\, \underbrace{(\forall \text{ bitstrings } \sigma \text{ of length } k)\, (\exists n < k)}_{\text{bounded number quantifiers}}\, \underbrace{R(\sigma \upharpoonright n, -)}_{\text{recursive}} \\
&\equiv \Sigma^0_1
\end{aligned}$$

□

Remark 48.4. Lemmas 48.1, 48.2, and 48.3 will be useful in Homework #11, Problems 1, 3, and 4. They will also be useful in the proof of the Low Basis Theorem, below.

Corollary 48.5. Let

$$P_e = \{X \in 2^{\mathbb{N}} \mid \varphi_e^{(1),X}(0)\uparrow\}, \qquad e = 0, 1, 2, \ldots$$

be our standard recursive enumeration of all Π^0_1 subsets of $2^{\mathbb{N}}$. Then, the set $\{e \mid P_e \neq \emptyset\}$ is Π^0_1. Also, the sets $\{3^i 5^j \mid P_i \cap P_j \neq \emptyset\}$, etc., are Π^0_1.

Proof. By the Magic Lemma 48.3 we have

$$P_e \neq \emptyset \;\equiv\; \underbrace{\exists X\, \underbrace{(X \in P_e)}_{\Pi^0_1}}_{\Pi^0_1}$$

and similarly

$$P_i \cap P_j \neq \emptyset \;\equiv\; \exists X\, (X \in P_i \wedge X \in P_j) \;\equiv\; \Pi^0_1$$

etc. □

49 The Low Basis Theorem

Definition 49.1. X is *low* if $X' \equiv_T 0'$. Note that this implies $X <_T 0'$.

Theorem 49.2 (Low Basis Theorem). Given a nonempty Π^0_1 set $P \subseteq 2^{\mathbb{N}}$, we can find an element $X \in P$ such that X is low.

Remark 49.3. The Low Basis Theorem should be compared with the result from Homework #8 that there is a nonempty Π^0_1 set $P \subseteq 2^{\mathbb{N}}$ with no recursive elements.

In general, a "basis theorem" is a theorem asserting that any nonempty "nice" set must contain a "nice" element. Thus, the low elements of $2^{\mathbb{N}}$ form a "basis" for the Π^0_1 sets, but the recursive elements of $2^{\mathbb{N}}$ do not.

Proof of the Low Basis Theorem. Let $P \subseteq 2^{\mathbb{N}}$ be nonempty Π^0_1. We shall obtain $X \in P$ by a technique known as "Π^0_1 approximation." This means that, starting with P, we shall construct a descending sequence of nonempty Π^0_1 sets

$$P = Q_0 \supseteq Q_1 \supseteq \cdots \supseteq Q_e \supseteq Q_{e+1} \supseteq \cdots$$

and then let $X \in \bigcap_{e=0}^{\infty} Q_e$. Note that $\bigcap_{e=0}^{\infty} Q_e$ is nonempty in view of Corollary 47.8.

We shall perform this construction in such a way as to insure that X is low, i.e., $H^X \leq_T H$ where H is the Halting Problem. Recall that $H^X = \{e \mid \varphi_e^{(1),X}(0)\downarrow\}$. Actually, the entire construction will be $\leq_T H$.

Here is the construction.

Stage 0: Let $Q_0 = P$.

Stage $e+1$: The purpose of this stage is to decide whether $e \in H^X$ or not. Assume inductively that Q_e is already known and is a nonempty Π^0_1 set.

Case 1: $(\exists X \in Q_e)\,(\varphi_e^{(1),X}(0)\uparrow)$. In this case let

$$Q_{e+1} = \{X \in Q_e \mid \varphi_e^{(1),X}(0)\uparrow\}$$

and note that Q_{e+1} is a nonempty Π^0_1 set. Moreover, $e \notin H^X$ for all $X \in Q_{e+1}$.

Case 2: Not case 1. I.e., $(\forall X \in Q_e)\,(\varphi_e^{(1),X}(0)\downarrow)$. In this case let $Q_{e+1} = Q_e$. Again Q_{e+1} is a nonempty Π^0_1 set. Moreover, $e \in H^X$ for all $X \in Q_{e+1}$.

The construction insures that $e \in H^X$ if and only if Case 2 holds at stage $e+1$. It remains to verify that the entire construction is $\leq_T H$. It will then follow that $H^X \leq_T H$, i.e., X is low.

Lecture 32: November 8, 2007

Here are the details of why the entire construction is $\leq_T H$.

We shall use our standard recursive enumeration P_e, $e = 0, 1, 2, \ldots$ of all Π^0_1 subsets of $2^{\mathbb{N}}$, namely

$$P_e = \{X \in 2^{\mathbb{N}} \mid \varphi_e^{(1),X}(0)\uparrow\}\,.$$

We say that e is an *index* of the Π^0_1 set P_e. Note also that $X' = H^X = \{e \mid \varphi_e^{(1),X}(0)\downarrow\} = \{e \mid X \notin P_e\}$.

Recall that if P, Q are Π^0_1 subsets of $2^{\mathbb{N}}$, then $P \cap Q$ is of course Π^0_1 and this holds *uniformly* with respect to the indices. In other words, we can find a recursive function $f(i,j)$ such that $P_{f(i,j)} = P_i \cap P_j$ for all i, j. (This is proved by means of the Parametrization Theorem.)

To see that the construction is $\leq_T H$, we shall define a function $g(e)$ and define Q_e as $Q_e = P_{g(e)}$. In other words, $g(e)$ will be an index of Q_e. To define g, let $B = \{3^i \cdot 5^j \mid P_i \cap P_j \neq \emptyset\}$. By Lemma 48.3, B is Π^0_1. See also Corollary 48.5. It follows that $B \leq_T 0'$.

Our construction may be described as follows. Start by letting $g(0)$ be an index of P. If $P_{g(e)} \cap P_e \neq \emptyset$, let $g(e+1) = f(g(e), e)$, so that $P_{g(e+1)} = P_g \cap P_e$. If $P_{g(e)} \cap P_e = \emptyset$, let $g(e+1) = g(e)$, so that $P_{g(e+1)} = P_{g(e)}$. Note that

$g \leq_T B \leq_T 0'$. Moreover, for all e, $e \in H^X$ if and only if $3^{g(e)}5^e \notin B$. Hence $X' = H^X \leq_T B \leq_T 0'$. Thus X is low. This completes the proof. □

We now apply the Low Basis Theorem to draw a conclusion about randomness. First, note the following:

Theorem 49.4. Let

$$R = \{X \in 2^{\mathbb{N}} \mid X \text{ is (Martin-Löf) random}\}.$$

Then R is Σ^0_2.

We present two proofs of this result.

First Proof. Our first proof is based on the existence of a universal test for randomness. Let V_n, $n = 0, 1, 2, \cdots$ be a universal test for randomness. Thus V_n is uniformly Σ^0_1, $\mu(V_n) \leq 1/2^n$ (a test), and $\forall X\,(X$ is random $\Leftrightarrow X$ passes the test, i.e., $X \notin \bigcap_{n=0}^{\infty} V_n)$. Then

$$R \;=\; \underbrace{2^{\mathbb{N}} \setminus \underbrace{\bigcap_{n=0}^{\infty} V_n}_{\Pi^0_2}}_{\Sigma^0_2} \;.$$

□

Second Proof. Our second proof is based on Schnorr's Theorem. We have

$$\begin{aligned} X \in R \quad &\equiv \quad K(X \upharpoonright n) \geq n - O(1) \\ &\equiv \quad \exists c\, \forall n\, (K(X \upharpoonright N) \geq n - c) \\ &\equiv \quad \underbrace{\exists c\; \underbrace{\forall n\, \forall \sigma\, [\underbrace{(U(\sigma) \simeq X \upharpoonright n)}_{\Sigma^0_1} \Rightarrow |\sigma| \geq n - c]}_{\Pi^0_1}}_{\Sigma^0_2} \;. \end{aligned}$$

□

Corollary 49.5. We can find a nonempty Π^0_1 set $P \subseteq 2^{\mathbb{N}}$ such that $\forall X\,(X \in P \Rightarrow X$ is random).

Proof. Since R is Σ^0_2, R is the union of a sequence of Π^0_1 sets. Since R is nonempty, at least one of these is nonempty. □

Remark 49.6.

1. We can actually find P as in the corollary such that $\mu(P) \geq 1 - \epsilon$ for any $\epsilon > 0$. This follows easily from the fact that $\mu(R) = 1$.

2. It can be shown that any P as in the corollary is Turing isomorphic to R (Kučera 1985).

Corollary 49.7. We can find $X \in 2^{\mathbb{N}}$ such that X is random and low.

Proof. This follows from the previous corollary plus the Low Basis Theorem. □

Similar to the Low Basis Theorem, there is the Hyperimmune-Free Basis Theorem:

Theorem 49.8 (Hyperimmune-Free Basis Theorem). If $P \subseteq 2^{\mathbb{N}}$ is Π^0_1, nonempty then $\exists X \in P\,(X$ is hyperimmune-free).

Proof. See Homework #11 Problem 4. The proof is by Π^0_1 approximation as in the proof of the Low Basis Theorem. Starting with P define a descending sequence of nonempty Π^0_1 sets

$$P = Q_0 \supseteq Q_1 \supseteq Q_2 \supseteq \cdots \supseteq Q_e \supseteq \cdots .$$

By compactness, $\bigcap_{n=0}^{\infty} Q_n$ is nonempty. Break the definition of hyperimmune-freeness into countably many requirements, and at stage $e+1$ construct Q_{e+1} to satisfy requirement e. The details are left to the student. □

Corollary 49.9. We can find $X \in 2^{\mathbb{N}}$ which is random and hyperimmune-free.

Proof. This follows from the Hyperimmune-Free Basis Theorem, just as the previous corollary followed from the Low Basis Theorem. □

Remark 49.10. We cannot combine these corollaries to get a random X which is both low and hyperimmune-free. In fact, the only Turing degree which is both low and hyperimmune-free is $\mathbf{0}$. See Homework #11, Problem 5.

Lecture 33: November 9, 2007

50 Randomness relative to an oracle

Recall that X is *random relative to* f (i.e., f*-random*) if $X \notin \bigcap_{n=0}^{\infty} V_n^f$, where V_n^f is any uniformly $\Sigma^{0,f}_1$ sequence of sets with $\mu(V_n^f) \leq 1/2^n$.

Lemma 50.1. Assume that $A \oplus B$ is random. Then A is B-random and B is A-random. In particular, A and B are random, and $A \not\leq_T B$ and $B \not\leq_T A$.

This lemma strengthens the result of Homework #9 Problem 4, which said that if X is random then $X_0 \not\leq_T X_1$ and $X_1 \not\leq_T X_0$. Here $X_0 =$ the even part of X and $X_1 =$ the odd part of X, defined by $X = X_0 \oplus X_1$.

Corollary 50.2. There is an infinite descending sequence of Turing degrees.

Proof. Let X be random and consider $X >_T X_0 >_T X_{00} >_T X_{000} >_T \cdots$. □

Proof of Lemma. Suppose B is not A-random. Then $B \in \bigcap_{n=0}^{\infty} V_n^A$ where V_n^A is uniformly $\Sigma_1^{0,A}$ and $\mu(V_n^A) \leq 1/2^n$. Letting $\Phi(X, Y, n)$ be a partial recursive functional such that $V_n^A = \{Y \mid \Phi(A, Y, n) \downarrow\}$, define $V_n^X = \{Y \mid \Phi(X, Y, n)\}$ and $W_n = \{X \oplus Y \mid Y \in V_n^X[1/2^n]\}$. Here we are using the isomorphism $2^{\mathbb{N}} \cong 2^{\mathbb{N}} \times 2^{\mathbb{N}}$ given by $X \oplus Y \mapsto (X, Y)$. Note that W_n is uniformly Σ_1^0, and by Fubini's Theorem $\mu(W_n) \leq 1/2^n$. Also $A \oplus B \in W_n$ because $B \in V_n^A = V_n^A[1/2^n]$ for all n. This contradicts the assumption that $A \oplus B$ is random. □

Theorem 50.3 (Van Lambalgen's Theorem). The following are pairwise equivalent:

1. $A \oplus B$ is random.

2. A is random and B is A-random.

3. B is random and A is B-random.

Proof. The previous lemma gives $1 \Rightarrow 2$ and $1 \Rightarrow 3$. We will prove $2 \Rightarrow 1$, and the proof of $3 \Rightarrow 1$ is similar. Assume $2 \wedge \neg 1$, i.e., A is random, B is A-random, and $A \oplus B$ is not random. Since $A \oplus B$ is not random, we have $A \oplus B \in \bigcap_{n=0}^{\infty} W_n$ where W_n is uniformly Σ_1^0 and $\mu(W_n) \leq 1/2^n$. By passing to a subsequence, we may assume that $\mu(W_n) \leq 1/2^{2n}$. Let $V_n^X = \{Y \mid X \oplus Y \in W_n\}$ and $U_n = \{X \mid \mu(V_n^X) > 1/2^n\}$. Note that U_n is uniformly Σ_1^0, because

$$\begin{aligned} X \in U_n \quad &\equiv \quad \mu(V_n^X) > 1/2^n \\ &\equiv \quad \exists s \; \underbrace{\mu(V_{n,s}^X) > 1/2^n}_{R(X,n,s) \text{ recursive}} \end{aligned}$$

Moreover, $\mu(U_n) \leq 1/2^n$ because otherwise by Fubini's Theorem we would have

$$\mu(W_n) \;\geq\; \mu(U_n) \cdot \frac{1}{2^n} \;>\; \frac{1}{2^n} \cdot \frac{1}{2^n} \;=\; \frac{1}{2^{2n}}$$

a contradiction. Since A is random, Solovay's Lemma tells us that $A \notin U_n$ for all but finitely many n. In other words, $\mu(V_n^A) \leq 1/2^n$ for all but finitely many n. But V_n^A is uniformly $\Sigma_1^{0,A}$, and $B \in V_n^A$ for all n (since $A \oplus B \in W_n$). Thus B is not A-random. This completes the proof. □

Theorem 50.4 (Miller/Yu 2004). Assume A is random and $A \leq_T B$ where B is C-random. Then A is C-random.

Proof. We omit the proof. □

51 Comments on Homework #11

In Homework #11, Problem 1 is to prove a generalization of the Magic Lemma 48.3, which states that the class of Π_1^0 predicates is closed under $\exists X$. Recall that the proof of the Magic Lemma used the fact that $2^{\mathbb{N}}$ is compact. Note that $2^{\mathbb{N}}$ is a product space,

$$2^{\mathbb{N}} \;=\; \{0,1\}^{\mathbb{N}} \;=\; \prod_{n=0}^{\infty}\{0,1\} \;=\; \{0,1\}\times\{0,1\}\times\{0,1\}\times\cdots\ .$$

There is a theorem of general topology known as Tychonoff's Theorem, which says that the product of any family of compact spaces is compact. Since $\{0,1\}$ is compact, we could have used Tychonoff's Theorem to prove that $2^{\mathbb{N}}$ is compact. Similarly, for any fixed $g \in \mathbb{N}^{\mathbb{N}}$ the product space

$$P_g \;=\; \prod_{n=0}^{\infty}\{0,1,2,\ldots,g(n)-1\} \;=\; \{f \in \mathbb{N}^{\mathbb{N}} \mid f \text{ is majorized by } g\}$$

is compact, by Tychonoff's Theorem. Another way to see this is to note that the full g-tree

$$T_g \;=\; \{\sigma \in \mathbb{N}^{<\mathbb{N}} \mid (\forall n < |\sigma|)\,(\sigma(n) < g(n))\}$$

is a finitely branching tree, so König's Lemma applies. This remark will be useful in solving Problem 1.

Lemmas 48.1, 48.2, 48.3 are useful in Homework #11. Also useful is the fact that, in $\mathbb{N}^{\mathbb{N}}$, Π^0_2 sets are Turing isomorphic to Π^0_1 sets. This was proved in a previous homework.

Lecture 34: November 12, 2007

Reminder: Professor Cholak, an expert on Kolmogorov complexity and randomness, will give two talks tomorrow, Tuesday November 13.

- MASS Seminar, 10:10–12:05.
- Logic Seminar, 2:30–3:45.

Problem 2(b)

Recall

$$0^{(n)} = 0^{\underbrace{\prime\prime\cdots\prime}_{n}} = \text{the } n\text{th Turing jump of } 0.$$

By Post's Theorem, the set $0^{(n)} \subseteq \mathbb{N}$ is Σ^0_n complete. Define

$$0^{(\omega)} = \bigoplus\nolimits_{n=0}^{\infty} 0^{(n)} = \{3^m 5^n \mid m \in 0^{(n)}\}.$$

Thus $0^{(\omega)}$ is not Σ^0_n for any n, i.e., it is not in the arithmetical hierarchy. We have seen in an earlier homework problem that the singleton set $\{0^{(\omega)}\}$ is Π^0_2. Consider the predicates

$$P(X,n) \;\equiv\; X = 0^{(\omega)} \,\wedge\, X(n) = 1$$

and

$$Q(n) \;\equiv\; \exists X\, P(X,n) \;\equiv\; n \in 0^{(\omega)}\ .$$

Note that while P is Π^0_2, Q is not arithmetical.

Problem 3

Let $P \subseteq 2^{\mathbb{N}}$ be Π^0_1. Let $\Phi(X,n)$ be a partial recursive functional such that $\Phi(X,n)\downarrow$ for all $X \in P$ and all n. We are asked to find a total recursive function $g(n)$ exceeding $\Phi(X,n)$ for all $X \in P$ and all n.

First we show that for each n, the values of $\Phi(X,n)$ for $X \in P$ are bounded. We have

$$\forall n\,(\forall X \in P)\,\exists i\,(\Phi(X,n) \simeq i)$$

hence

$$\forall n\,\forall X\,\exists i\,(\underbrace{X \notin P}_{\Sigma^0_1} \vee \underbrace{\Phi(X,n) \simeq i}_{\Sigma^0_1})$$

hence by Lemma 48.2 (the bounding principle)

$$\forall n\,\exists j\,\forall X\,(\exists i < j)(X \notin P \vee \Phi(X,n) \simeq i)$$

hence

$$\forall n\,\exists j\,(\forall X \in P)\,(\exists i < j)\,(\Phi(X,n) \simeq i)$$

so the values are bounded. Now consider the predicate $Q(n,j)$ saying that j is an appropriate bound, i.e.,

$$\begin{aligned} Q(n,j) &\equiv (\forall X \in P)\,(\exists i < j)\,(\Phi(X,n) \simeq i) \\ &\equiv \forall X\,(\exists i < j)\,(X \notin P \vee \Phi(X,n) \simeq i) \\ &\equiv \underbrace{\forall X\,(\underbrace{X \notin P \vee \Phi(X,n)\downarrow < j}_{\Sigma^0_1})}_{\Sigma^0_1} \end{aligned}$$

by Lemma 48.3. Thus the predicate $Q(n,j)$ is Σ^0_1. Since $\forall n\,\exists j\,Q(n,j)$ holds, we can find a recursive function $g(n)$ such that $\forall n\,Q(n,g(n))$. This is the g as desired.

52 Homework #12, due November 26, 2007

Exercises 52.1.

1. (a) Let P be a Π^0_1 subset of $2^{\mathbb{N}}$. If P has only finitely many elements, prove that all of the elements of P are recursive.

   Hint: Use Lemma 48.3, a.k.a., the Magic Lemma.

   (b) Does this hold with $\mathbb{N}^{\mathbb{N}}$ instead of $2^{\mathbb{N}}$?

2. (a) Let $P \subseteq 2^{\mathbb{N}}$ be nonempty Π^0_1 with no recursive elements. Prove that for all Y we can find $X \in P$ such that $X' \equiv_T X \oplus 0' \equiv_T Y \oplus 0'$.

   Note: This result is a combination of the Low Basis Theorem and the Friedberg Jump Inversion Theorem. The proof is basically a combination of the two proofs.

(b) Deduce that for all Y we can find a random X such that $X' \equiv_T X \oplus 0' \equiv_T Y \oplus 0'$.

3. Recall that we have defined

$$K(n) = K(\langle \underbrace{1, \ldots, 1}_{n} \rangle)$$

for all $n \in \mathbb{N}$.

Assume that $f(n)$ is a recursive function such that

$$\sum_{n=0}^{\infty} \frac{1}{2^{f(n)}} \; < \; \infty \, .$$

Prove that $K(n) \leq f(n) + O(1)$ for all n.

4. Prove that

$$K(\tau) \leq C(\tau) + K(C(\tau)) + O(1)$$

for all bitstrings τ.

53 The Kučera/Gács Theorem

Continuing the line already pursued in Sections 46 and 49, we now present another theorem about $\deg_T(X)$ where X is random in the sense of Martin-Löf.

Theorem 53.1 (Kučera 1985). For all $Y \geq_T 0'$ we can find X such that $X \equiv_T Y$ and X is random.

Corollary 53.2 (Gács). $\forall Y \, \exists X \, (X$ random and $Y \leq_T X)$.

Corollary 53.3. $\exists X \, (X$ random and $0' \leq_T X)$.

To prove the theorem, we first prove some lemmas.

Lemma 53.4. Let $P \subseteq 2^{\mathbb{N}}$ be a measurable set. Let σ be a bitstring such that $\mu(P \cap N_\sigma) \geq 1/2^k$ where $k \geq 1$. Then we can find at least two distinct bitstrings τ of length $2k$ extending σ such that $\mu(P \cap N_\tau) \geq 1/2^{4k}$.

Proof. Note first that

$$\frac{1}{2^{|\sigma|}} \; = \; \mu(N_\sigma) \; \geq \; \mu(P \cap N_\sigma) \; \geq \; \frac{1}{2^k}$$

hence $|\sigma| \leq k < 2k$ since $k \geq 1$. It follows that

$$P \cap N_\sigma \; = \; \bigcup_{\tau \supset \sigma, |\tau| = 2k} P \cap N_\tau \quad \text{(disjoint union)}$$

hence

$$\mu(P \cap N_\sigma) \ = \sum_{\tau \supset \sigma, |\tau|=2k} \mu(P \cap N_\tau) \ .$$

Suppose for a contradiction that there is at most one τ as required. Then

$$\sum_{\tau \supset \sigma, |\tau|=2k} \mu(P \cap N_\tau) \leq \frac{1}{2^{2k}} + (2^{2k-|\sigma|} - 1)\frac{1}{2^{4k}} < \frac{1}{2^{2k}} + 2^{2k}\frac{1}{2^{4k}} = \frac{2}{2^{2k}} \leq \frac{1}{2^k}$$

so $\mu(P \cap N_\sigma) < 1/2^k$ a contradiction. □

Lecture 35: November 14, 2007

The following lemma has been implicit in previous results and homework, but we pause to make it explicit.

Lemma 53.5. Let $P \subseteq 2^{\mathbb{N}}$ be Π^0_1. Then, the 2-place number-theoretic predicate

$$Q(\tau, k) \quad \equiv \quad \mu(P \cap N_\tau) \geq \frac{1}{2^k}$$

is Π^0_1.

Proof. Let $T \subseteq 2^{<\mathbb{N}}$ be a recursive tree such that $P = \{\text{paths through } T\}$. Then

$$\mu(P \cap N_\tau) \geq \frac{1}{2^k} \quad \equiv \quad (\forall n \geq |\tau|) \left(\frac{|\{\sigma \in T \mid \sigma \supseteq \tau, |\sigma| = n\}|}{2^n} \geq \frac{1}{2^k} \right)$$

and this is clearly Π^0_1. □

Lemma 53.6. Let $P \subseteq 2^{\mathbb{N}}$ be Π^0_1 of positive measure. Then for all Y we can find $X \in P$ such that $Y \leq_T X$. Moreover $X \leq_T Y \oplus 0'$.

Proof. The idea of the proof is to apply Lemma 53.4 repeatedly to construct a mapping $f : 2^{<\mathbb{N}} \to 2^{<\mathbb{N}}$. Namely, for each $\rho \in 2^{<\mathbb{N}}$, $f(\rho^\frown\langle 0\rangle)$ and $f(\rho^\frown\langle 1\rangle)$ will be incompatible bitstrings extending $f(\rho)$ obtained from Lemma 53.4. For technical reasons, we choose $f(\rho^\frown\langle 0\rangle)$ and $f(\rho^\frown\langle 1\rangle)$ to be the leftmost and rightmost such strings. Here "leftmost" and "rightmost" are with respect to some fixed recursive linear ordering of all bitstrings.

Here are the details of the construction of f. Fix $k \geq 1$ such that $\mu(P) \geq 1/2^k$. Define $f(\rho)$ by induction on $|\rho|$. Begin with $f(\langle\rangle) = \langle\rangle$. Note that $N_{\langle\rangle} = 2^{\mathbb{N}}$ hence $\mu(P \cap N_{\langle\rangle}) = \mu(P) \geq 1/2^k$. Assume inductively that $f(\rho)$ has already been defined and $\mu(P \cap N_{f(\rho)}) \geq 1/2^{4^n k}$ where $n = |\rho|$. By Lemma 53.4 there are at least two bitstrings τ extending $f(\rho)$ of length $2 \cdot 4^n k$ such that $\mu(P \cap N_\tau) \geq 1/2^{4^{n+1}k}$. Let $f(\rho^\frown\langle 0\rangle)$ and $f(\rho^\frown\langle 1\rangle)$ be the leftmost and rightmost such τ. Note that $f(\rho^\frown\langle 0\rangle)$ and $f(\rho^\frown\langle 1\rangle)$ are distinct bitstrings of length $2 \cdot 4^n k$.

By Lemma 53.5 we have $f \leq_T 0'$. Given $Y \in 2^{\mathbb{N}}$, let $X = \bigcup_{n=0}^\infty f(Y \upharpoonright n)$. Clearly $X \leq_T Y \oplus f \leq_T Y \oplus 0'$ and $Y \leq_T X \oplus f \leq_T X \oplus 0'$. Moreover,

$X \in N_{f(Y\restriction n)}$ and $P \cap N_{f(Y\restriction n)} \neq \emptyset$ for all n. Since P is a closed set, it follows that $X \in P$.

It remains to show that $Y \leq_T X$. Using X as an oracle, we describe how to compute Y. Suppose we have already computed $Y \restriction n$. We need to decide whether $Y(n) = 0$ or $Y(n) = 1$. We know what $f(Y \restriction n+1)$ is, namely it is $X \restriction 2 \cdot 4^n k$. We also know what $f(Y \restriction n)$ is, namely it is $X \restriction 2 \cdot 4^{n-1}k$ if $n > 0$, or $\langle\rangle$ if $n = 0$. Moreover, we know that $Y(n) = 0$ (respectively $Y(n) = 1$) if and only if $\mu(P \cap N_\tau) < 1/2^{4^{n+1}k}$ for all τ extending $f(Y \restriction n)$ of length $2 \cdot 4^n k$ lying to the left (respectively right) of $X \restriction 2 \cdot 4^n k$. By Lemma 53.5 these predicates are Σ^0_1. Therefore, since one of these predicates holds, we can wait until we find out which one holds, and at that point we know whether $Y(n) = 0$ or $Y(n) = 1$. This completes the proof. □

Lecture 36: November 15, 2007

Proof of Theorem 53.1. Let P be a Π^0_1 set containing only random X. For any Turing degree $\mathbf{a} \geq \mathbf{0}'$, let Y be such that $\mathbf{a} = \deg_T(Y)$. Then by the above lemma, we can find $X \in P$ such that $Y \leq_T X$ and $X \leq Y \oplus \mathbf{0}'$. But $Y \oplus \mathbf{0}' \equiv_T Y$, so $X \equiv_T Y$. □

In addition Jonas Kibelbek presented solutions of Problems 7 and 8 in Homework #11. See Section 55 below.

Lecture 37: November 23, 2007

Jonas presented solutions of some problems in Homework #10. See Section 54 below.

Lecture 38: November 26, 2007

We have proved the following two theorems.

Theorem 53.7 (van Lambalgen's Theorem). The following are pairwise equivalent:

1. $A \oplus B$ is random

2. A is random, and B is random relative to A.

3. B is random, and A is random relative to B.

Theorem 53.8 (Kučera/Gács Theorem). For all $Y \geq_T 0'$ we can find a random $X \equiv_T Y$.

We now combine these two theorems to deduce the following corollaries.

Corollary 53.9. Suppose A is random, $A \leq_T B$, B is random relative to C, and $C \geq_T 0'$. Then A is random relative to C.

Proof. Since $C \geq_T 0'$, we can assume by Kučera/Gács that C is random. Since B is random relative to C, it follows by van Lambalgen that $B \oplus C$ is random. Hence, by van Lambalgen again, C is random relative to B. Hence, since $A \leq_T B$, C is random relative to A. We are also assuming that A is random, so by van Lambalgen we get that $A \oplus C$ is random. Applying van Lambalgen again, we see that A is random relative to C. □

Remark 53.10. The previous corollary actually holds without the assumption $C \geq_T 0'$. This result is due to Miller/Yu 2004.

Definition 53.11. We say that A is *n-random* if A is random relative to $0^{(n-1)}$. We say that A is *arithmetically random* if A is n-random for all n.

For example,

$$\begin{aligned} \text{1-random} &\equiv \text{random}, \\ \text{2-random} &\equiv \text{random relative to } 0', \\ \text{3-random} &\equiv \text{random relative to } 0'', \\ \text{etc.} \end{aligned}$$

Thus we have a hierarchy which we can use to measure higher and higher amounts of randomness.

Corollary 53.12. Assume that A is random and $A \leq_T B$.

1. If B is n-random, then so is A.
2. If B is arithmetically random, then so is A.

Proof. In the previous corollary, let $C = 0^{(n-1)}$. □

Remark 53.13. The above corollaries can be paraphrased as follows:

> If A is random, and if $A \leq_T B$ for some B which is "highly random," then A itself is "highly random."

Thus we see that Martin-Löf randomness is in a sense a minimal threshhold of randomness. Beyond this threshhold, higher amounts of randomness behave nicely in that they propagate downward via Turing reducibility. This phenomenon may be viewed as further evidence for our belief that Martin-Löf's concept of randomness is a very natural concept.

54 Some solutions for Homework #10

Problem 4

Let $X \in 2^{\mathbb{N}}$. We say that X is *2-random* if X is random relative to $0'$. Recall also that X is *weakly 2-random* if $X \notin$ any Π^0_2 set of measure 0. Let $\mathbf{a} = \deg_T(X) =$ the Turing degree of X.

Part (a): Show that if X is 2-random then X is weakly 2-random.

Solution. We need to show

$$(X \notin \text{ any Martin-Löf test relative to } 0') \Rightarrow \big(X \notin \text{ any measure } 0\ \Pi^0_2 \text{ set}\big)$$

We will do this by showing that every measure 0 Π^0_2 set P is also a Martin-Löf test relative to $0'$.

Since P is Π^0_2, we have $X \in P \equiv \forall m\, \exists n\, R(X, m, n)$ where R is recursive. If we consider the Σ^0_1 sets $W_m = \{X \mid \exists n\, R(X, m, n)\}$, we see that $P = \displaystyle\bigcap_{m=0}^{\infty} W_m$. That is, a Π^0_2 set is the intersection of a uniform sequence of Σ^0_1 sets. We may assume that $W_0 \supseteq W_1 \supseteq \cdots \supseteq W_m \supseteq \cdots$, since we could replace each W_m with $\tilde{W}_m = \displaystyle\bigcap_{i=0}^{m} W_i$, which is again a uniform sequence of Σ^0_1 sets.

The difference between P and a Martin-Löf test is that we do not have a nice bound on how quickly $\mu(W_m)$ goes to 0. (Recall that a Martin-Löf test is $T = \bigcap_{n=0}^{\infty} V_n$ where the V_n are uniformly Σ^0_1 and $\mu(V_n) \leq \frac{1}{2^n}$.) We can use the Halting Problem to estimate the measures $\mu(W_m)$ and take a subsequence $f(n)$ so that $\mu(W_{f(n)}) \leq 1/2^n$.

Recall that we have a standard way to index Σ^0_1 sets; $U_e = \{X \mid \varphi_e^{(1),X}(0)\downarrow\}$. By the Parametrization Theorem, we can find a total function $h(m)$ such that $W_m = U_{h(m)}$. It is helpful to define the sets $W_{m,s} = U_{h(m),s} = \{X \mid \varphi_{e,s}^{(1),X\upharpoonright s}(0)\downarrow\}$, which are "finite approximations of W_m." For all s, $W_{m,s}$ is recursive, and $W_{m,0} \subseteq W_{m,1} \subseteq \cdots \subseteq W_{m,s} \subseteq \cdots$ with $\displaystyle\bigcup_{s=0}^{\infty} W_{m,s} = W_m$.

Define the partial recursive function $\psi(n, m) =$ the least s such that $\mu(W_{m,s}) > 1/2^n$. Then $\psi(n, m)$ halts if $\mu(W_m) > 1/2^n$ and it fails to halt if $\mu(W_m) \leq 1/2^n$. So, using the Halting Problem, we can compute an m sufficiently large that $\mu(W_m) \leq 1/2^n$.

Define $f(n) =$ the least m such that $\psi(n, m)$ fails to halt. Then $W_{f(n)}$ is a uniform $\Sigma_1^{0,0'}$ sequence with $\mu(W_{f(n)}) \leq 1/2^n$. Thus, $P = \bigcap_{m=0}^{\infty} W_m = \bigcap_{n=0}^{\infty} W_{f(n)}$ is a Martin-Löf test relative to $0'$; and so every 2-random X is also weakly 2-random. □

Part (b): Show that if X is weakly 2-random then $\inf(\mathbf{a}, \mathbf{0}') = \mathbf{0}$.

Solution. Suppose that $\inf(\mathbf{a}, \mathbf{0}') \neq \mathbf{0}$. Let Y be such that $0 <_T Y \leq_T 0'$ and $Y \leq_T X$. Since $Y \leq_T 0'$, by Post's Theorem, Y is Δ^0_2. Since $Y \leq_T X$, there is some e such that $Y = \varphi_e^{(1),X}$.

We will show that the set $P = \{X \mid Y = \varphi_e^{(1),X}\}$ is Π^0_2 of measure 0, so that X cannot be weakly 2-random. Since Y is nonrecursive, we know that P has measure 0. (See problem 7 of Homework #11.) We check that P is Π^0_2:

$$X \in P \equiv \underbrace{\forall n \underbrace{\left(\underbrace{\varphi_e^{(1),X}(n)\downarrow}_{\Sigma^0_1} \wedge \underbrace{\varphi_e^{(1),X}(n) = Y(n)}_{\Delta^0_2}\right)}_{\Delta^0_2}}_{\Pi^0_2}$$

□

Part (c): In Part (b), what if we assume only that X is random?

Solution. By the Low Basis Theorem, we can show that there exists X that are random and low. That is, $X' \equiv_T \mathbf{0}'$, which implies $X <_T \mathbf{0}'$. Then if $\mathbf{a} = \deg_T(X)$, $\inf(\mathbf{a}, \mathbf{0}') = \mathbf{a} \neq \mathbf{0}$. Thus, the result of part (b) does not hold if we only assume that X is random. □

Part (d): If X is 2-random then $X' \equiv_T X \oplus 0'$.

Solution. Recall that $X' = H^X = \{e \mid \varphi_e^{(1),X}(0)\downarrow\}$ = the Halting Problem relative to X.

Let U_e = $\{X \mid \varphi_e^{(1),X}(0)\downarrow\}$
= the eth Σ^0_1 subset of $2^{\mathbb{N}}$.

Let $U_{e,s}$ = $\{X \mid \varphi_{e,s}^{(1),X\restriction s}(0)\downarrow\}$
= $\bigcup_{\sigma\in 2^{\mathbb{N}},|\sigma|=s} N_\sigma$ where $\varphi_{e,s}^{(1),\sigma}(0)\downarrow$.

Note that

1. $U_{e,s}$ is a finite union of neighborhoods.
2. $\mu(U_{e,s})$ is a recursive function of e, s.
3. $U_e = \displaystyle\bigcup_{s=0}^{\infty} U_{e,s}$, hence $\mu(U_e) = \lim_{s\to\infty}\mu(U_{e,s})$.

Let $f(e)$ = the least s such that $\mu(U_e \setminus U_{e,s}) \leq 1/2^e$. We claim that $f \leq_T 0'$. This is because $f(e)$ = the least s such that $(\forall t \geq s)\,(\mu(U_{e,t} \setminus U_{e,s}) \leq 1/2^e)$ which is a Π^0_1 condition.

Let $V_e = U_e \setminus U_{e,f(e)}$. Note that $\mu(V_e) \leq 1/2^e$. Moreover V_e is $\Sigma^{0,f}_1$, hence $\Sigma^{0,0'}_1$. So the sets V_e, $e = 0, 1, 2, \ldots$ form a test for randomness relative to $0'$.

By Solovay's Lemma relative to $0'$, since X is random relative to $0'$, $X \notin V_e$ for all but finitely many e.

Therefore, for all sufficiently large e, $e \in H^X \equiv X \in U_e \equiv X \in U_{e,f(e)}$. It follows that $H^X \leq_T X \oplus f \leq_T X \oplus 0'$, which is what we need to show that $X' \equiv_T X \oplus 0'$. □

55 Some solutions for Homework #11

Problem 3

Let $P \subseteq 2^{\mathbb{N}}$ be Π^0_1. Let $\Phi(X, n)$ be a partial recursive functional such that $\Phi(X, n) \downarrow$ for all $X \in P$ and all n. We are to find a recursive upper bound on these values, $g(n)$, where g is a total recursive function.

One solution was given in class earlier, using several lemmas. A more hands-on approach is to note that for each $X \in P$ and each n there is a finite amount of information from X that causes $\Phi(X, n)$ to halt. This finite amount of information is essentially a neighborhood, and P is covered by these neighborhoods. Since P is a Π^0_1 subset of $2^{\mathbb{N}}$, it is closed and hence compact, so we can find a finite subcovering. In fact, we can find such a finite subcovering recursively. Since we now have only finitely many neighborhoods to consider, we can take $g(n)$ to be one more than the maximum value of $\Phi(X, n)$ on these neighborhoods.

Problem 4

Given $P \subseteq 2^{\mathbb{N}}$ Π^0_1 nonempty. To find a hyperimmune-free $X \in P$, we construct a descending sequence of nonempty Π^0_1 sets

$$P = Q_0 \supseteq Q_1 \supseteq \cdots \supseteq Q_e \supseteq Q_{e+1} \supseteq \cdots$$

and at the end of the construction we let $X \in \bigcap_{e=0}^{\infty} Q_e$.

Stage 0. Let $Q_0 = P$.

Stage $e + 1$. Given Q_e, there are two cases.

Case 1: $\exists n \, (\exists X \in Q_e) \, (\varphi_e^{(1),X}(n) \uparrow)$. In this case, fix such an n and let $Q_{e+1} = \{X \in Q_e \mid \varphi_e^{(1),X}(n) \uparrow\}$. Note that Q_{e+1} is Π^0_1 because we have chosen a particular n and defined

$$X \in Q_{e+1} \;\; \equiv \;\; X \in Q_e \wedge \varphi_e^{(1),X}(n) \uparrow \;\; .$$

Moreover Q_{e+1} is nonempty by the choice of n. We have now satisfied the eth requirement for hyperimmune-freeness, because the partial X-recursive function $\varphi_e^{(1),X}(n)$ is not total for any $X \in Q_{e+1}$.

Case 2: Not case 1. In this case let $Q_{e+1} = Q_e$. Then, the partial recursive functional $\Phi(X, n) \simeq \varphi_e^{(1),X}(n)$ is $\downarrow$ for all $X \in Q_{e+1}$ and all n. Hence by Problem 3 we can find a total recursive function g_e such that $\varphi_e^{(1),X}(n) \downarrow < g_e(n)$ for all $X \in Q_{e+1}$ and all n. Thus, in this case also, the eth requirement for hyperimmune-freeness has been satisfied.

Problem 7

Part (a): Prove that if Y is nonrecursive then $\mu(\{X \mid Y \not\leq_T X\}) = 1$.

Solution. This is equivalent to saying that $\mu(\{X \mid Y \leq_T X\}) = 0$. We can simplify by defining the sets $C_e = \{X \mid Y = \varphi_e^{(1),X}\}$. Then our problem is equivalent to showing that $\mu(C_e) = 0$ for all e. (Since the union of countably many measure 0 sets has measure 0.)

Suppose for some e, C_e has positive measure. It would be nice if we could conclude that there is some neighborhood $N_\sigma \subseteq C_e$, so that Y can be computed, using the program given by e, using as an oracle any $X \supset \sigma$. This would make Y recursive, since it could be computed using just the finite amount of information in σ, which would give us the needed contradiction.

However, it is not true that if C_e has positive measure, it must contain some neighborhood. (For example, the set of random elements has measure 1, but contains no neighborhoods.) We can however approximate C_e closely by neighborhoods. By regularity of the measure μ, there is an open set U containing C_e with $\mu(U) < 4\mu(C_e)/3$. Since U is an open set, it is a union of neighborhoods, $U = \bigcup_{\sigma \in S} N_\sigma$. We can take a finite subset F of S so that $V = \bigcup_{\sigma \in F} N_\sigma$ has measure $\mu(V) > 3\mu(U)/4$. This gives us a very simple open set V that is a good approximation of C_e. The set V is a finite union of neighborhoods, and at least $2/3$ of V by measure is in C_e. (The worst case is that all of $U \setminus C_e$ is in V, but $\mu(U \setminus C_e) < \mu(U)/4 < \mu(V)/3$.)

This means that, by measure, most of the $X \in V$ are such that $\varphi_e^{(1),X} = Y$. So, to compute $Y(n)$, find the least k such that for strings σ of length k with $N_\sigma \subseteq V$, the computation $\varphi_e^{(1),\sigma}(n)$ halts with the same value for enough σ's to account for at least $1/3$ of the measure of V. Then the value at which they halt is $Y(n)$. Thus, Y is computable, which contradicts our initial assumption.

Hence, $C_e = \{X \in 2^{\mathbb{N}} \mid Y = \varphi_e^{(1),X}\}$ has measure 0 for all e. □

Part (b): Deduce that for each nonrecursive Y we can find a random X such that $Y \not\leq_T X$.

Solution. Note that the set $R = \{X \mid X \text{ is random}\}$ has measure 1. So, for any nonrecursive Y, $R \cap \{X \in 2^{\mathbb{N}} \mid Y \not\leq_T X\}$ is the intersection of two measure 1 sets, and thus has measure 1 and is nonempty. □

Part (c): More generally, given a sequence of nonrecursive oracles Y_i, $i = 0, 1, 2, \ldots$, find an X which is n-random for all n such that $Y_i \not\leq_T X$ for all i.

Solution. For each n, the set $R_n = \{X \mid X \text{ is } n\text{-random}\}$ has measure 1. (This is because the Martin-Löf tests relative to $0^{(n-1)}$ can be enumerated as T_{ni}, $i = 0, 1, 2, \ldots$, i.e., there are only countably many of them. Then $R_n = 2^{\mathbb{N}} \setminus \bigcup_{i=0}^{\infty} T_{n,i}$ where $\bigcup_{i=0}^{\infty} T_{ni}$ has measure 0 because each $T_{n,i}$ has measure 0.)

Thus $\bigcap_{n=0}^{\infty} R_n$ has measure 1. Similarly, $\bigcap_{i=0}^{\infty}\{X \in 2^{\mathbb{N}} \mid Y_i \not\leq_T X\}$ has measure 1. Their intersection still has measure 1 and so is nonempty. □

Problem 8

Part (a): Assume that $P \subseteq 2^{\mathbb{N}}$ is Π_1^0 and $\neg \exists X \, (X \in P \wedge X \text{ is recursive})$. Find a nonrecursive Y such that $\nexists X \, (X \in P \wedge X \leq_T Y)$.

Solution. We will construct such a Y by finite approximation. We need to guarantee that Y is nonrecursive. (This step is straightforward; we have done it several times before.) The new step is to guarantee that Y cannot be used to compute any $X \in P$. That is, for each e, we need to guarantee that $\varphi_e^{(1),Y}$ is not in P.

Stage 0. Let $\sigma_0 = \langle\rangle$.

Stage 2e+1.

Case 1: If $\varphi_e(|\sigma_{2e}|) = 0$, let $\sigma_{2e+1} = \sigma_{2e}{}^\frown\langle 1\rangle$.

Case 2: If $\varphi_e(|\sigma_{2e}|) \neq 0$, let $\sigma_{2e+1} = \sigma_{2e}{}^\frown\langle 0\rangle$.

The odd numbered steps guarantee that $Y = \bigcup_{n=0}^{\infty} \sigma_n$ is nonrecursive.

Stage 2e+2. Find a $\sigma \supset \sigma_{2e+1}$ such that there is no $Y \supset \sigma$ with $\varphi_e^{(1),Y}$ in P. Let $\sigma_{2e+2} = \sigma$.

Claim: There is no need for a case 2, since such a σ always exists.

It is clear that if the claim is true, then $Y = \bigcup_{n=0}^{\infty} \sigma_n$ will not compute any X in P, and the problem will be complete.

Suppose there is no such σ. Then for all $\tau \supseteq \sigma_{2e+1}$, there is some $Y \supset \tau$ with $\varphi_e^{(1),Y}$ in P. We will show that some X in P is recursive.

To compute X, first find the least $\tau_0 \supseteq \sigma_{2e+1}$ such that $\varphi_e^{(1),\tau_0}(0)$ halts; define $X(0) = \varphi_e^{(1),\tau_0}(0)$.

Given τ_n, let τ_{n+1} be the least extension of τ_n such that $\varphi_e^{(1),\tau_n}(n)$ halts; define $X(n+1) = \varphi_e^{(1),\tau_{n+1}}(n+1)$.

Note that we can always find such extensions, by the assumption that every string τ extending σ_{2e+1} is an initial segment for some Y where $\varphi_e^{(1),Y}$ is total and in P. Clearly, X is recursive. Since P is Π_1^0, P is the set of infinite paths through a recursive tree T. Since each τ_n is an initial segment for some Y with $\varphi_e^{(1),Y}$ in P, $\varphi_e^{(1),\tau_n} \upharpoonright n = \varphi_e^{(1),Y} \upharpoonright n$ is in the tree T. That is, $X \upharpoonright n = \varphi_e^{(1),\tau_n} \upharpoonright n$ is in T for all n; thus $X \in P$.

This gives us our contradiction, since we assume that P contains no recursive elements. □

Part (b): Find a nonrecursive Y such that $\neg \exists X\,(X \text{ is random} \wedge X \leq_T Y)$.

Solution. Recall that the set of random elements $R = 2^{\mathbb{N}} \setminus \bigcap_{n=0}^{\infty} V_n$ where $\bigcap_{n=0}^{\infty} V_n$ is a universal Martin-Löf test and each V_n is Σ_1^0. Then $R = \bigcup_{n=0}^{\infty}(2^{\mathbb{N}} \setminus V_n)$ is the union of countably many Π_1^0 sets.

So, to construct a Y that does not compute any random X, we modify the above finite approximation construction so that we take care of all the sets $P_n = 2^{\mathbb{N}} \setminus V_n$. (Note that these are Π_1^0 sets without any recursive elements, as required.)

Stage 0: Let $\sigma_0 = \langle\rangle$.

Stage 2k+1:

Case 1: If $\varphi_k(|\sigma_{2k}|) = 0$, let $\sigma_{2k+1} = \sigma_{2k}{}^\frown\langle 1\rangle$.

Case 2: If $\varphi_k(|\sigma_{2k}|) \neq 0$, let $\sigma_{2k+1} = \sigma_{2k}{}^\frown\langle 0\rangle$.

The odd numbered steps guarantee that $Y = \bigcup_{n=0}^{\infty} \sigma_n$ is nonrecursive.

Stage 2k+2: Let $n = (k)_0$ and $e = (k)_1$. Find a $\sigma \supset \sigma_{2k+1}$ such that there is no $Y \supset \sigma$ with $\varphi_e^{(1),Y}$ in P_n. Let $\sigma_{2k+2} = \sigma$.

Let $Y = \bigcup_{n=0}^{\infty} \sigma_n$. Then Y is nonrecursive and no $X \in P_n$ for any n is Turing reducible to Y. Since each random $X \in$ some P_n, no random X is Turing reducible to Y. □

56 Comments on Homework #12

Problem 4

The problem should have been to prove $K(\tau) \leq C(\tau) + K(C(\tau)) + O(1)$.

Solution. Define M by

$$M(\rho^\frown\sigma) \simeq \tau \quad \equiv \quad U(\rho) = \langle \underbrace{1, \ldots, 1}_{|\sigma|} \rangle \wedge U^*(\sigma) \simeq \tau$$

where U is a universal prefix-free machine and U^* is a universal machine. It is easy to verify that M is a prefix-free machine. Given τ, let σ be such that $U^*(\sigma) \simeq \tau$ and $|\sigma| = C(\tau)$. Let ρ be such that $U(\rho) = \langle \underbrace{1, \ldots, 1}_{|\sigma|} \rangle$ and $|\rho| = K(|\sigma|)$. Then $M(\rho^\frown\sigma) \simeq \tau$, hence $K(\tau) \leq |\rho| + |\sigma| + O(1) = C(\tau) + K(C(\tau)) + O(1)$. □

57 Homework #13, due December 3, 2007

Exercises 57.1. Recall that A is said to be *strongly random* if A does not belong to any Π_2^0 set of measure 0.

1. (a) Suppose $A \oplus B$ is strongly random. Prove that $\inf(\mathbf{a}, \mathbf{b}) = \mathbf{0}$ where $\mathbf{a} = \deg_T(A)$ and $\mathbf{b} = \deg_T(B)$.

   (b) What if we assume only that $A \oplus B$ is random?

2. Prove the following. If A is random and $A \leq_T B$ and B is strongly random, then A is strongly random.

3. Prove that the following conditions are equivalent.

   (a) A is random relative to $0^{(n)}$ for all n.

   (b) A does not belong to any arithmetical set of measure 0.

   In this case we say that A is *arithmetically random.*

58 LR-reducibility

We shall end the course by discussing some additional methods of classifying Turing oracles. These classification methods are motivated by the ideas of relative randomness and relative prefix-free complexity, respectively.

Definition 58.1 (LR-reducibility). Write $A \leq_{LR} B$ to mean that

$$\forall X \text{ (if } X \text{ is } B\text{-random then } X \text{ is } A\text{-random).}$$

Write $A \equiv_{LR} B$ to mean that $A \leq_{LR} B$ and $B \leq_{LR} A$.

Remark 58.2. Clearly $A \leq_T B$ implies $A \leq_{LR} B$. Moreover, LR-reducibility is similar to Turing reducibility in that it is a transitive, reflexive relation on Turing oracles. Similarly, $A \equiv_T B$ implies $A \equiv_{LR} B$, and $\equiv_{LR}$ is similar to $\equiv_T$ in that it is an equivalence relation.

The following theorem says that LR-reducibility does not coincide with Turing reducibility. The idea of LR-reducibility is that we are classifying oracles according to their ability to reveal nonrandom patterns.

Theorem 58.3 (Kučera/Terwijn 2002). For all A we can find B such that $A <_T B$ and $A \equiv_{LR} B$.

Theorem 58.4 (Kjos-Hanssen 2005). $A \leq_{LR} B$ if and only if every $\Pi^{0,A}_1$ set of positive measure includes a $\Pi^{0,B}_1$ set of positive measure.

Definition 58.5 (LK-reducibility). Write $A \leq_{LK} B$ to mean that

$$K^B(\tau) \quad \leq \quad K^A(\tau) + O(1)$$

for all bitstrings τ. Write $A \equiv_{LK} B$ to mean that $A \leq_{LK} B$ and $B \leq_{LK} A$.

Remark 58.6. Just as for LR-reducibility, we have similar properties for LK-reducibility. Namely, $A \leq_T B$ implies $A \leq_{LK} B$, and LK-reducibility is reflexive and transitive, and $\equiv_{LK}$ is an equivalence relation.

The idea of LK-reducibility is that we are classifying oracles according to their ability to compress bitstrings. From this point of view, the following theorem is interesting and remarkable. It says that LR-reducibility and LK-reducibility coincide.

Theorem 58.7 (Kjos-Hanssen/Miller/Solomon 2006). $A \leq_{LR} B$ if and only if $A \leq_{LK} B$.

In the time remaining, we shall try to prove as many of these theorems as possible.

Lecture 39: November 28, 2007

Remark 58.8 (Final Examinations). December 10, 12, 14. Each student will have an individual 1-hour oral final exam. The exam will consist of a question from the list handed out in class, a problem to solve similar to easier homework questions, and a presentation of the research project. Please turn in the paper for your research project by **8 am on Monday, December 10** so that it can be reviewed before the final examination, when grades are assigned.

We now begin the proofs of some of the theorems on LR-reducibility.

Let U, V be Σ^0_1 subsets of $2^{\mathbb{N}}$, with $U = \bigcup_{\sigma \in S} N_\sigma$, $V = \bigcup_{\tau \in T} N_\tau$ where S, T are prefix-free Σ^0_1 subsets of $2^{<\mathbb{N}}$.

Define a product operation $UV = \bigcup_{\sigma \in S} \bigcup_{\tau \in T} N_{\sigma ^\frown \tau}$.

Properties:

1. UV is Σ^0_1.
2. Given indices of U, V (qua Σ^0_1 sets), we can compute an index of UV (qua Σ^0_1 set). Namely,
$$\left(\begin{array}{l} U_e = \{X \mid \varphi_e^{(1),X}(0)\downarrow\} = \bigcup\limits_{\sigma \in S_e} N_\sigma \\ \text{where } S_e = \{\sigma \mid \varphi_{e,|\sigma|}^{(1),\sigma}(0)\downarrow \wedge (\forall \rho \subset \sigma)\varphi_{e,|\rho|}^{(1),\rho}(0)\uparrow\} \end{array} \right)$$
3. $UV \subseteq U$ (because $N_{\sigma ^\frown \tau} \subseteq N_\sigma$).
4. $\mu(UV) = \mu(U)\mu(V)$.

   (because each N_σ is a copy of the entire Cantor space, $N_\sigma V$ has measure $\mu(V)/2^{|\sigma|}$, and $UV = \bigcup_{\sigma \in S} N_\sigma V$.)
5. The product is associative. $(UV)W = U(VW)$.

Define $U^n = \underbrace{U \cdots U}_{n}$. Then $\mu(U^n) = \mu(U)^n$. If $\mu(U) < 1$, then $\lim_{n\to\infty} \mu(U^n) = 0$. Let k be such that $\mu(U^k) \leq 1/2$, then $\mu(U^{nk}) = (\mu(U^k)^n) \leq 1/2^n$, hence U^{nk}, $n = 0, 1, 2, \ldots$ is a Martin-Löf test.

Lecture 40: November 29, 2007

Review:

Definition 58.9. Let $U, V \subseteq 2^{\mathbb{N}}$ be open with $U = \bigcup_{\sigma \in S} N_\sigma$ and $V = \bigcup_{\tau \in T} N_\tau$ where S, T are prefix-free sets of bitstrings. Define $UV = \bigcup_{\sigma \in S, \tau \in T} N_{\sigma ^\frown \tau}$. This is again an open set.

Remark 58.10. This product operation UV is not really an operation on open sets. Rather, it is an operation on the prefix-free sets of bitstrings which define these open sets. To be absolutely correct we should write $ST = \{\sigma ^\frown \tau \mid \sigma \in S, \tau \in T\}$ and note that this is again a prefix-free set of bitstrings. However, we shall instead continue to abuse notation by writing UV as if it were an operation on open sets.

Remark 58.11. Our product operation UV has the following properties:

1. $UV \subseteq U$.
2. $(UV)W = U(VW)$.
3. $\mu(UV) = \mu(U)\mu(V)$.
4. If U and V are Σ^0_1, then UV is Σ^0_1. Moreover, this holds uniformly.

Letting $U^n = \underbrace{U \cdots U}_{n}$ we see that $\mu(U^n) = \mu(U)^n$. Therefore, if $\mu(U) < 1$ we have $\lim_{n\to\infty} \mu(U^n) = \lim_{n\to\infty} \mu(U)^n = 0$ geometrically. If in addition U is Σ^0_1, then the sequence of sets U^n, $n = 0, 1, 2, \ldots$ is uniformly Σ^0_1, so we have a Martin-Löf test. Thus, for any random $X \in 2^{\mathbb{N}}$ we have $X \notin U^n$ for some n. Consider the least such n. Then $X \in U^{n-1}$ and $X \notin U^n = U^{n-1}U$. Thus $X = \sigma_1{}^\frown \cdots {}^\frown\sigma_{n-1}{}^\frown Y$ for some $\sigma_1, \ldots, \sigma_{n-1} \in S$ and some $Y \notin U$.

We have now essentially proved the following lemma:

Lemma 58.12 (Kučera, 1985). Let $P \subseteq 2^{\mathbb{N}}$ be Π^0_1 of positive measure. Then for all random $X \in 2^{\mathbb{N}}$ there exist σ and Y such that $X = \sigma^\frown Y$ and $Y \in P$.

Proof. Let $U = 2^{\mathbb{N}} \setminus P$ and reason as above. Let $\sigma = \sigma_1{}^\frown \cdots {}^\frown\sigma_{n-1}$ and let Y be such that $X = \sigma^\frown Y$. □

Remark 58.13. The above lemma is a refinement of the 0−1 Law in probability theory.

Corollary 58.14. Let P, Q be nonempty Π^0_1 sets consisting of random elements. Then P, Q are Turing isomorphic.

Proof. For each $X \in P$ we have $X = \sigma^\frown Y$ for some bitstring σ and some $Y \in Q$, hence $X \equiv_T Y$. And vice versa. □

The following lemma is implicit in earlier work.

Lemma 58.15. Let $P \subseteq 2^{\mathbb{N}}$ be Π^0_1. The following are pairwise equivalent:

1. $\mu(P) > 0$
2. P includes a nonempty Π^0_1 set consisting of random element.
3. P contains a random element.

Proof. ($1 \Rightarrow 2$) Recall that $R = \{X \mid X \text{ is random }\}$ is Σ^0_2, hence $R = \bigcup_{n=0}^{\infty} P_n$ where P_n is Π^0_1. Also, $\mu(R) = 1$. If $\mu(P) > 0$, then $\mu(P \cap P_n) > 0$ for some n. Thus $P \cap P_n$ is a Π^0_1 set which is included in P and consists entirely of random elements.

($2 \Rightarrow 3$) is trivial.

($3 \Rightarrow 1$) because random ⇒ weakly random. □

Our goal now is to prove the following result giving several characterizations of *LR*-reducibility.

Theorem 58.16 (Kjos-Hanssen 2005). The following are pairwise equivalent.

1. $A \leq_{LR} B$. (This means that B-random implies A-random.)
2. Every $\Pi^{0,A}_1$ set of positive measure includes a $\Pi^{0,B}_1$ set of positive measure.
3. There exists a $\Pi^{0,A}_1$ set consisting of A-random elements which includes a $\Pi^{0,B}_1$ set of positive measure.
4. There exists a $\Pi^{0,B}_1$ set of positive measure consisting of A-random elements.

Toward the proof of this theorem, note that $2 \Rightarrow 3$ follows from the previous lemma relativized to A. Also, $3 \Rightarrow 4$ is trivial. To prove $4 \Rightarrow 1$, assume 4 and let Q be a $\Pi^{0,B}_1$ set of positive measure consisting of A-random elements. If X is B-random, then by Lemma 58.12 relativized to B we have $X = \sigma^\frown Y$ for some $Y \in Q$. It follows that Y is A-random, hence X is A-random, and this proves 1.

It remains to prove $1 \Rightarrow 2$. In order to prove $1 \Rightarrow 2$ we make the following definition.

Definition 58.17. Let $U \subseteq 2^{\mathbb{N}}$ be open. We say that U is *fat* if U intersects every Π^0_1 set of positive measure. We say that U is *B-fat* if U intersects every $\Pi^{0,B}_1$ set of positive measure.

Note that 2 amounts to saying that every B-fat $\Sigma^{0,A}_1$ set is of measure 1.

Lemma 58.18. If U and V are fat, then UV is fat.

Proof. Let U and V be fat. Let Q be a Π^0_1 set of positive measure. By the previous lemma, we may assume every element of Q is random. Write $U = \bigcup_{\sigma \in S} N_\sigma$ where S is prefix-free. Then $UV = \bigcup_{\sigma \in S} \sigma^\frown V$. Since U is fat, $U \cap Q \neq \emptyset$, hence $N_\sigma \cap Q \neq \emptyset$ for some $\sigma \in S$. Hence $N_\sigma \cap Q$ is Π^0_1 of positive measure (by previous lemma). Since V is fat, $\sigma^\frown V$ is fat within N_σ. Hence $Q \cap (\sigma^\frown V) \neq \emptyset$, hence $Q \cap UV \neq \emptyset$, Q.E.D. □

Note also that Lemma 58.18 relativizes as follows:

For all B, if U and V are B-fat then UV is B-fat.

Lecture 41: November 30, 2007

Review:

Lemma 58.19. Every Π^0_1 set of positive measure includes a Π^0_1 set of positive measure consisting of random elements.

Proof. This is because $\{X \mid X \text{ is random}\}$ is Σ^0_2 of measure 1. □

Lemma 58.20. Every Π^0_1 set which contains a random element is of positive measure.

Proof. This is because randomness implies weak randomness. □

Lemma 58.21 (Kučera's Lemma). If P is Π^0_1 of positive measure then $\forall X$ (X random $\Rightarrow P$ contains a "tail" of X), i.e., $X = \sigma^\frown Y$ where $Y \in P$.

We have also proved:

Lemma 58.22. If U and V are open and fat, then UV is open and fat.
(Recall *fat* = "intersects every Π^0_1 set of positive measure.")

We shall now prove Theorem 58.16.

Proof. ($2 \Rightarrow 3$) is immediate from Lemma 58.19 relativized to A.

($3 \Rightarrow 4$) is trivial.

($4 \Rightarrow 1$) follows from Kučera's Lemma 58.21. Namely, let P be as in 4, i.e., P is $\Pi^{0,B}_1$ of positive measure consisting of A-random elements. Let X be B-random. By Kučera's Lemma, X has a tail in P, i.e., $X = \sigma^\frown Y$ where $Y \in P$. By assumption about P, Y is A-random. Therefore X is A-random.

It remains to prove ($1 \Rightarrow 2$).

Assume 1, i.e., $A \leq_{LR} B$. To prove 2, it suffices to prove that every B-fat $\Sigma^{0,A}_1$ set is of measure 1.

Let U be a $\Sigma^{0,A}_1$ set which is B-fat. Assume for a contradiction that $\mu(U) < 1$. As we have seen, the sets U^n, $n = 1, 2, 3, \ldots$ form a test for A-randomness. Hence every $X \in \bigcap_{n=1}^\infty U^n$ is not A-random. Also, since U is B-fat, by Lemma 58.22 relativized to B we see that U^n is B-fat for all n.

By Lemma 58.19 relativized to B, let P be nonempty $\Pi^{0,B}_1$ consisting of B-random elements. By Lemma 58.20 we have $\mu(P) > 0$. More generally, by Lemma 58.20 we have

$$\forall \sigma \, (P \cap N_\sigma \neq \emptyset \Rightarrow \mu(P \cap N_\sigma) > 0).$$

Since U^n is B-fat, it follows that

$$\forall n \, \forall \sigma \, (P \cap N_\sigma \neq \emptyset \Rightarrow (P \cap N_\sigma \cap U^n \neq \emptyset)).$$

Since U^n is open, it follows that

$$\forall n \, \forall \sigma \, (P \cap N_\sigma \neq \emptyset \Rightarrow (\exists \tau \supset \sigma) \, (P \cap N_\tau \neq \emptyset \wedge N_\tau \subseteq U^n)).$$

Apply this repeatedly starting with $\sigma_0 = \langle\rangle$ to get an increasing sequence of bitstrings $\sigma_0 \subset \sigma_1 \subset \cdots \subset \sigma_n \subset \sigma_{n+1} \subset \cdots$ such that for all n,

$$P \cap N_{\sigma_n} \neq \emptyset \text{ and } N_{\sigma_n} \subseteq U^n.$$

Finally let $X = \bigcup_{n=1}^\infty \sigma_n$. Then $X \in P$ (because P is closed), hence X is B-random. On the other hand $X \in \bigcap_{n=1}^\infty U^n$, hence X is not A-random. This contradicts our assumption $A \leq_{LR} B$.

This completes the proof of Theorem 58.16. □

We now prove one more important theorem concerning LR-reducibility.

Theorem 58.23 (Kučera/Terwijn, 2002). We can find a simple r.e. set A such that $A \leq_{LR} 0$.

Corollary 58.24. $\exists A\,(A \leq_{LR} 0$ and A not recursive).

Corollary 58.25. $\leq_{LR}$ does not coincide with $\leq_T$.

Some other known results are as follows:

Theorem 58.26. $A \leq_{LR} 0 \Rightarrow A$ is *low* i.e., $A' \equiv_T 0'$.

Corollary 58.27. $A \leq_{LR} 0 \Rightarrow A \leq_T 0'$, hence A is Δ^0_2.

Corollary 58.28. There are only countably many A such that $A \leq_{LR} 0$.

Theorem 58.29. If $A \leq_{LR} 0$ and $B \leq_{LR} 0$ then $A \oplus B \leq_{LR} 0$.

Theorem 58.30. $A \leq_{LR} 0 \Rightarrow \exists B\,(B$ r.e., $A \leq_T B$, $B \leq_{LR} 0)$.

Theorem 58.31. $A \leq_{LR} B \Leftrightarrow A \leq_{LK} B$.

Remark 58.32. Let A be a Turing oracle. If $A \leq_{LR} 0$ we say that A is *low-for-random.* If $A \leq_{LK} 0$ we say that A is *low-for-K*. The above results have been proved in the past few years and give much insight concerning oracles which are low-for-random. In particular, A is low-for-random if and only if A is low-for-K.

We now begin the proof of the Kučera/Terwijn Theorem 58.23.

Proof of Theorem 58.23. We know we can find a Π^0_1 set P with $\mu(P) > 1/2$ such that $\forall X\,(X \in P \Rightarrow X$ is random).

Let us uniformly relativize this to an arbitrary oracle C. Thus P^C is uniformly $\Pi^{0,C}_1$, $\mu(P^C) > 1/2$ and $\forall X\,(X \in P^C \Rightarrow X$ is C-random).

Let $U^C = 2^{\mathbb{N}} \setminus P^C$. Note that U^C is $\Sigma^{0,C}_1$ (uniformly) and $\mu(U^C) < 1/2$.

To prove the theorem, it will suffice to build a simple r.e. set A and a Σ^0_1 set V such that $U^A \subseteq V$ and $\mu(V) < 1$.

Then, letting $Q = 2^{\mathbb{N}} \setminus V$, it follows that Q is Π^0_1 and $\mu(Q) > 0$ and all elements of Q are A-random. Hence by Theorem 58.16 every random X is A-random, i.e., $A \leq_{LR} 0$, Q.E.D.

The proof will be completed next class. □

Lecture 42: December 3, 2007

We have reduced Theorem 58.23 to a lemma:

Lemma 58.33. For all oracles C let U^C be uniformly $\Sigma^{0,C}_1$ of measure $< 1/2$. Then we can find a simple r.e. set A and a Σ^0_1 set V such that $U^A \subseteq V$ and $\mu(V) < 1$.

We shall now prove this lemma and thereby complete the proof of Theorem 58.23.

Proof. We shall build A as $A = \bigcup_{s=0}^{\infty} A_s$ where $A_0 \subseteq A_1 \subseteq \cdots \subseteq A_s \subseteq A_{s+1} \subseteq \cdots$ is an increasing recursive sequence of finite sets. The entire construction will be recursive. Consequently A will be r.e. We will let $V = \bigcup_{s=0}^{\infty} U_s^{A_s \upharpoonright s}$ so that V is Σ_1^0. We need to insure (1) A is simple, (2) $\mu(V) < 1$.

Explanation: U^C is uniformly $\Sigma_1^{0,C}$, so let e be such that $U^C = \{X \mid \varphi_e^{(1),C\oplus X}(0)\downarrow\}$ for all oracles C. Define $U_s^{C\upharpoonright s} = \{X \mid \varphi_{e,s}^{(1),C\upharpoonright s\oplus X\upharpoonright s}(0)\downarrow\}$ and note that this is a finite union of neighborhoods, namely

$$U_s^{C\upharpoonright s} = \bigcup_{\substack{|\sigma| = s \\ \varphi_{e,s}^{(1),C\upharpoonright s\oplus\sigma}(0)\downarrow}} N_\sigma \quad .$$

In particular, $U_s^{A_s\upharpoonright s}$, $s = 0, 1, 2, \ldots$ is the union of a finite set of neighborhoods, and this finite set of neighborhoods can be computed recursively from s.

To control $\mu(V)$, let us write V as a disjoint union, $V = \bigcup_{t=0}^{\infty} V_t$ where $V_t = U_t^{A_t\upharpoonright t} \setminus \bigcup_{s<t} U_s^{A_s\upharpoonright s}$. Note that V_t, $t = 0, 1, 2, \ldots$ is again a recursive sequence of finite unions of neighborhoods. Moreover, the sets V_t, $t = 0, 1, 2, \ldots$ are pairwise disjoint.

Recall that an r.e. set A is said to be *simple* if (1) $\mathbb{N} \setminus A$ is infinite, (2) $\forall e\,(W_e \text{ infinite } \Rightarrow A \cap W_e \neq \emptyset)$. To insure that A is simple, we use our standard enumeration of all r.e. sets, $W_e = \{n \mid \varphi_e^{(1)}(n)\downarrow\}$. We use the approximations $W_{e,s} = \{n < s \mid \varphi_{e,s}^{(1)}(n)\downarrow\}$. Clearly $W_{e,s}$ is a recursive sequence of finite sets, and $W_e = \bigcup_{s=0}^{\infty} W_{e,s}$.

The construction of A is as follows.

Stage 0: Let $A_0 = \emptyset$.

Assume that A_t, $t = 0, 1, \ldots, s$ have already been defined. Hence $U_t^{A_t\upharpoonright t}$ and V_t, $t = 0, 1, \ldots, s$ have already been defined. In preparation for stage $s + 1$, define $c(n, s) = \mu(\bigcup_{n<t\leq s} V_t) = \sum_{n<t\leq s} \mu(V_t)$. The recursive function $c(n, s)$ is known as a "cost function." Its purpose is to measure the potential "cost" of putting n into A at stage $s + 1$.

Stage $s + 1$: For each $e < s$ such that $A_s \cap W_{e,s} = \emptyset$, look for $n \in W_{e,s}$ such that $n \geq 2e$ and $c(n, s) \leq 1/2^{e+2}$ and, if such an n is found, put the least such n into A_{s+1}.

This completes our description of the construction. Note that the entire construction is recursive.

We claim that $\mathbb{N} \setminus A$ is infinite.

To see this, note that for each e at most one n was put into A for the sake of intersecting W_e. Therefore, our restriction $n \geq 2e$ insures that $|A \cap \{0, 1, 2, \ldots, 2e-1\}| \leq e$ for all e. It follows that the complement of A is infinite.

We claim that W_e infinite $\Rightarrow A \cap W_e \neq \emptyset$.

To see this, note that $\sum_t \mu(V_t) = \mu(\bigcup_t V_t) \leq 1$, hence $\sum_{n<t} \mu(V_t) \to 0$ as $n \to \infty$. Since W_e is infinite, let $n \in W_e$ be so large that $n \geq 2e$ and $\sum_{n<t} \mu(V_t) \leq 1/2^{e+1}$. It follows that $c(n, s) = \sum_{n<t\leq s} \mu(V_t) \leq 1/2^{e+1}$ for all s. Let s be so large that $n \in W_{e,s}$. Then by construction $A_{s+1} \cap W_{e,s} \neq \emptyset$.

We claim that $\mu(V) < 1$.

To see this, it suffices to prove that $\mu(V \setminus U^A) \leq 1/2$ (because we already know that $\mu(U^A) < 1/2$). Given $X \in V \setminus U^A$, consider the unique t such that $X \in V_t$. Then $X \in U_t^{A_t \upharpoonright t} \setminus U^A$. It follows that $A \upharpoonright t \neq A_t \upharpoonright t$. Therefore, at some stage $s + 1 > t$, some $n < t$ must have been put into A_{s+1} for the sake of W_e for some $e < s$. For this particular e, the set of all such X's is included in $\bigcup_{n<t\leq s} V_t$ and is therefore of measure $\leq \sum_{n<t\leq s} \mu(V_t) = c(n, s) \leq 1/2^{e+1}$. Hence, the set of all such X's for all e is of measure $\leq \sum_{e=0}^{\infty} 1/2^{e+1} = 1/2$. This proves the claim.

The proof of Lemma 58.33 and Theorem 58.23 is now complete. □

59 A final word

I hope everyone has enjoyed this course and come away with a good understanding of computability, unsolvability, and randomness.