ASSIGNMENT 16: CHAITIN'S WORK

MICHAEL BEESON

1. Suppose that N is algorithmically random. In this problem we write log for the log to the base 2. Let

$$N = p_1^{e_1} \dots p_n^{e_n}$$

be the prime factorization of N.

(a) Show that $H(N) = \log N$. Show that the information content $H(N)$ is at most a constant plus the sum of the $H(e_i)$ for $i = 1$ to n.

(b) Show that $e_i \leq \log N$, and hence $H(e_i) \leq \log \log N$.

(c) Derive a lower bound on the number n of prime factors of N.

(d) Conclude that the existence of an algorithmically random N implies that there are infinitely many primes. Since this is a non-trivial theorem, then the existence of algorithmically random N must also be considered non-trivial.

2. Alice and Bob are, for a change, cooperating. Bob has to determine exactly which programs of size N terminate (without input). Alice is allowed to give him N bits of information. There are at least two ways to choose the N bits so that Bob (knowing what the bits mean) can correctly make his predictions. Your task is to explain how Bob will use the information from Alice in each of the two cases.

(a) Alice can tell Bob how many programs of size N halt. (So the N bits are just the bits of this number; there are at most 2^N programs of size N, so the number fits in N bits.)

(b) Alice can tell Bob which program takes the longest to halt. (So the N bits are just the longest-running, but terminating, program of size N.)

(c) Conclude that an $N + c$-bit axiom system *can* settle all questions of the form "$\varphi_e(0)$ halts" for e of size up to N, where c is a constant independent of N.

3. In this exercise you will prove the existence of an algorithmically random bit string (or number) of length N. You can't do this with Turing machines as usually defined. Instead, we assume we have some programming language that is "prefix-free", which means no initial segment of a program is a program. (Turing machines are not prefix-free.)

(a) How many numbers of size exactly N are there? (Careful, they can't begin with 0. For example, there are just `10` and `11` of size 2.)

(b) Show that, because of the prefix-free property, there are not as many programs of size strictly less than N as there are numbers of size exactly N. *Hint*: What is the sum of the first k powers of 2? For example $1 + 2 + 4 + 8 = 15$.

(c) conclude that there must be some number of size N that is not the output of a program with fewer than N bits. (That is the desired algorithmically random string.)

4. (a) Explain how to modify the definition of Turing machine so it *would* have the prefix-free property. (Hence the preceding exercise applies to the so-modified Turing machines.)

(b) State a version of the existence of algorithmically random strings that applies to the (unmodified) notion of Turing machine computation.