CS5000 HW10

Chase Mortensen A01535275

November 2019

1

Show that there is a computable predicate $P(x_1, ..., x_n, t)$ such that $min_t P(x_1, ..., x_n, t)$ is not computable.

In order for a function to be computable, it must be total and be able to be expressed as an L-program. The main issue with $min_t P(x_1, ..., x_n, t)$ is that t is the last value passed to the function. If no upper bound is passed to the function then the range is infinite and the function is undefined.

2

A prime number p is a larger twin prime if p and $p2$ are both primes. For example, 5 is a larger twin prime, because $5 - 2 = 3$ is a prime; 7 is a larger twin prime, because $7 - 2 = 5$ is also a prime. 13 is a larger twin prime, because $13 - 2 = 11$ is also a prime; 19 is a larger prime, because $19 - 2 = 17$ is a prime. One of the current conjectures in number theory, which has not been proved yet, is that there are infinitely many twin primes, i.e., pairs of natural numbers p and $p2$ that are both primes. Let's assume for the sake of this problem that this conjecture is true. Let $T(0) = 0$ and let $T(n)$ be the n-th larger twin prime. Show that $T(n)$ is computable.

$T(n)$ is computable because it is entirely composed of computable parts. In order for a function to be computable, it must be both total and be able to be expressed as an L-program. Because there are theoretically infinitely many twin primes, $T(n)$ is total. I won't write an L-program for $T(n)$, but the program would consist of computable parts, such as $prime(x)$, addition, loops, etc. Since it is composed of computable parts, $T(n)$ is also computable.

3

A function is *elementary* if it can be obtained from the functions $s(x)$, $n(x)$, u_i^j, $+$, $\dot{-}$ by a finite number of applications of composition, bounded summation, and bounded product. Let $f(t, x_1, ..., x_n)$ is a function and let $g(y, x_1, ..., x_n) = \Sigma_{t=0}^{y} f(t, x_1, ..., x_n)$. Then g is obtained from f by application of bounded summation. Let $h(y, x_1, ..., x_n) = \Pi_{t=0}^{y} f(t, x_1, ..., x_n)$. Then h is obtained from f by application of bounded product.

1. Show that every elementary function is primitive recursive.

   Elementary functions are primitive recursive because they are derived from "a finite number of applications of composition, bounded summation, and bounded product" which are obtained from $s(x)$, $n(x)$, u_i^j, $+$, $-$. Since those have all been shown to be primitive recursive, the elementary functions are primitive recursive.

2. Show that $x \times y$, x^y, $x!$ are elementary.

   For $x \times y$, a bounded summation from 0 to x of a function that returns a constant y will yield the correct result. x^y can be obtained in a similar way: a bounded product from 0 to x of a function that returns a constant y. $x!$ can be represented as a bounded product of a function that returns the iterator.

3. Show that if $n+1$-ary predicates P and Q are elementary, then so are $\neg P$, PvQ, $P\&Q$, $(\forall t)_{\leq y} P(t, x_1, ..., x_n)$, $(\exists t)_{\leq y} P(t, x_1, ..., x_n)$, and $min(t)_{\leq y} P(t, x_1, ..., x_n)$.

   The negation, or, and, for all, existential, and minimization operations are all elementary because they can derived from "a finite number of applications of composition, bounded summation, and bounded product". First, the negation operation could be obtained using alpha, which would qualify because it can be derived from the functions $s(x)$, $n(x)$, u_i^j, $+$, $\dot{-}$ Or can be represented with $+$. And can be represented as $(Q(x) + P(x) \dot{-} 1$. The for all function is essentially a bounded product and the existential operator is bounded summation. Minimization can be represented as a bounded product inside of a bounded summation. So, all of the operators are elementary if the predicates are elementary.

4. Show that the predicate $Prime(x)$ that returns true if x is a prime is elementary.

$Prime(x)$ is elementary. It can be represented entirely with elementary components, so it is elementary. $Prime(x)$ can be represented using a bounded summation of a function that checks if the iterator is a divisor of x. The $gcd(i, x)$ function would have to return 1 on every number less than x. The gcd function can also be represented as elementary, along with all of the other pieces, so $Prime(x)$ is elementary.

4

1. Let $<< x_1, x_2 >, < x_3, x_4 >>= 1223$. Solve for x_1, x_2, x_3, x_4.

   I'm assuming we are using $2^x(2y + 1) - 1$ since there is no pairing method specified. 1223 can be split into $< 3, 76 >$. $(2^3(2(76) + 1) - 1 = 1223$. 3 can be split into $< 2, 0 >$. $(2^2(2(0) + 1) - 1 = 3)$. 76 can be split into $< 0, 38 >$. $(2^0(2(38) + 1) - 1$. So, $x_1 = 2$, $x_2 = 0$, $x_3 = 0$, and $x_4 = 38$.

2. Let $<< x_1, < x_2, x_3 >>, < x_4, x_5 >>= 50011$. Solve for x_1, x_2, x_3, x_4, x_5.

   I'm kind of questioning whether or not this was the intended pairing function, but I am still using $2^x(2y + 1) - 1$. So, $l(50011) = 2$ and $r(50011) = 6251$. So, $x_1 = l(l(50011)) = 0$, $x_2 = l(r(l(50011))) = 1$, $x_3 = r(r(l(50011))) = 0$, $x_4 = l(r(50011)) = 2$, and $x_5 = r(r(50011)) = 781$.