

**Master Parisien de Recherche en Informatique**  
**Theory of Computations**

---

Assignment 6: Solutions

Handed out November 6, 2011

---

**Solution 1.**

Let  $A$  be a  $\Pi_2^0$  set. By definition we can write it as

$$A = \{n \mid \forall x \exists y P(n, x, y)\}$$

where  $P$  is a total computable predicate. Let then  $F : \mathbb{N}^2 \rightarrow \{0, 1\}$  be defined by the following algorithm with parameter  $n$ :

$x \leftarrow 0$

$y \leftarrow 0$

for  $t = 0$  to  $\infty$

If  $P(n, x, y) = 0$ , set  $F(n, t) = 0$ .

If  $P(n, x, y) = 1$ , set  $F(n, t) = 1$ , then  $x \leftarrow x + 1$  and  $y \leftarrow 0$

In other words, the algorithm goes through all values of  $x$  in order. For each  $x$  it tries to find a  $y$  such that  $P(n, x, y) = 1$ , and outputs 0's while waiting to find one. If for all  $x$  there is a  $y$  such that  $P(n, x, y) = 1$ , then  $F(n, \cdot)$  will output infinitely many 1's. Otherwise,  $x$  eventually reaches a value  $x_0$  for which there is no  $y$  such that  $P(n, x_0, y) = 1$ , in which case the function  $F(n, \cdot)$  eventually stabilizes on 0. Thus

$$n \in A \Leftrightarrow F(n, t) = 1 \text{ for infinitely many } t$$

The converse is easy: if there is such a function  $F$  for a set  $A$ , then

$$A = \{n \mid \forall x \exists y (y \geq x) \wedge (F(n, y) = 1)\}$$

so  $A$  is  $\Pi_2^0$ .

**Solution 2.**

One can write  $X = \{e \mid \forall n \forall t \exists t' > t (n \in A \Rightarrow n \in W_e[t'])\}$ , and the predicate " $(n \in A \Rightarrow n \in W_e[t'])$ " is  $A$ -computable, hence  $X$  is  $\Pi_2^0[A]$ . To see that  $X$  computes  $A$ , we many-one reduce  $A$  to  $X$  as follows. Let  $f$  be the total computable function such that  $W_{f(e)} = \mathbb{N} \setminus \{e\}$  for all  $e$  (to prove that such an  $f$  exists, proceed as usual, applying the s-m-n theorem to the function  $\phi_a(x, y)$  which is defined if and only if  $x \neq y$ ). Then  $f$  is a many-one reduction of  $A$  to  $X$  as  $n \notin A \Leftrightarrow A \subseteq \mathbb{N} \setminus \{e\}$ , i.e.,  $n \in A \Leftrightarrow f(n) \in X$ .

To get  $X \equiv_T A$ , take  $A = \emptyset$ . This makes  $X = \mathbb{N}$ , so  $A \equiv_T X$ . For  $X \equiv_T A'$ , take  $A = \{0\}$ . Then  $A \equiv_T \emptyset$ , and  $X = \{e \mid \phi_e(0) \downarrow\}$ , which as we have seen in the lectures is  $\Sigma_1^0$ -complete. So  $X \equiv_T A'$ . Finally, to get  $X \equiv_T A''$ , take  $A = \mathbb{N}$ . In this case,  $X = \text{Tot}$  which as we know is  $\Pi_2^0$ -complete, hence Turing-equivalent to  $\emptyset''$ .

**Solution 3.**

(a) Let us consider the partial computable function suggested in the hint:

$$f(n) = \min\{t \mid \phi_n(n)[t] \downarrow\}$$

Clearly  $f$  is partial computable. Suppose it is extended by a total computable function  $g$ . Then

- either  $\phi_n(n)[g(n)] \downarrow$  (when  $f$  is defined  $g$  is equal to  $f$ )
- or  $\phi_n(n) \uparrow$

Either way, we have  $\phi_n(n) \downarrow \Leftrightarrow \phi_n(n)[g(n)] \downarrow$ , but the predicate on the right hand side is computable, which would make  $\emptyset'$  computable, a contradiction.

(b) This is trivial. If the domain  $D$  of the function  $f$  is computable, let  $g$  be the function which on input  $n$  first checks whether  $n \in D$ , in which case it outputs  $f(n)$ , and outputs 0 otherwise.

(c) One can write

$$\text{Ext} = \{e \mid \exists i (i \in \text{Tot}) \wedge \forall n \forall t \exists t' > t \phi_e(n)[t'] \downarrow \Rightarrow \phi_i(n)[t'] = \phi_e(n)[t']\}$$

Both predicates  $(i \in \text{Tot})$  and  $\forall n \forall t \exists t' > t \phi_e(n)[t'] \downarrow \Rightarrow \phi_i(n)[t'] = \phi_e(n)[t']$  are  $\Pi_2^0$  in  $i$ . With the extra  $\exists i$ , we see that  $\text{Ext}$  is  $\Sigma_3^0$ .

(d) We follow the hint. We use a construction as in the proof of the completeness of  $\text{Cof}$ . That is, given a  $\Sigma_3^0$  set  $A$  which we want to reduce to  $\text{Ext}$ , we write

$$A = \{n \mid \exists x \forall y \exists z P(n, x, y, z)\}$$

where  $P$  is a computable predicate. On parameter  $n$ , we build a partial computable function  $f$ . As before, the marker number  $x$  makes an active move (pushing the ones of higher number) for the  $(y + 1)$ -th time if we find a  $z$  such that  $P(n, x, y, z) = 1$ . Markers are originally "active". When a marker number  $x$  is active and positioned on an integer  $k$ , it runs the computation of  $\phi_x(k)$  and if this computation halts before the marker moves, we set  $f(n) = \phi_x(k) + 1$  (ensuring  $f \neq \phi_x$ ). The marker number  $x$  then becomes inactive forever (though it still moves as usual) and makes an extra passive move. When a position  $k$  is left without a marker, if  $f$  is not yet defined at  $k$ , set  $f(n) = 42$ . This construction defines a partial computable function  $f$  (we have an algorithm which assigns values to  $f$  in an enumerable way). One thing one needs to check is that an active marker is never on a position where  $f$  is already defined. Indeed, when an active marker number  $x$  causes  $f(n)$  to be defined on position  $k$ , then the position is immediately vacated. This means that at the end of each step, markers are precisely on the positions where  $f$  is undefined.

Now, there are two scenarios:

- Either some marker moves infinitely often. In this case, as for  $\text{Cof}$ , we show that almost all positions are eventually left without a marker. Since  $f$  is defined

on any such position,  $f$  is then defined for all but finitely many integers. This in particular means that  $f$  has computable domain, so by (b)  $f$  can be extended to a computable function.

- Or no marker moves infinitely often. In this case, for all  $x$ , the marker  $x$  reaches a final position. Then for each  $x$ ,  $\phi_x$  is not a total computable extension of  $f$ , because:
  - either the marker  $x$  was already inactive when it arrived on this final position. Then by construction this means that at some earlier position it ensured  $f(k) \neq \phi_x(k)$
  - or the marker was active when it arrives at its final position  $k$ . Then we know that  $\phi_x(k)$  will never be defined, otherwise the active marker would define  $f(k)$  and then moves, contradicting that  $k$  is its final position. Thus  $\phi_x$  is not defined at  $k$ .

In both cases,  $\phi_x$  is not an extension of  $f$  (in one case because  $f$  and  $\phi_x$  differ at some value, in the other case because  $\phi_x$  is not total).

This shows that  $f$  is extendible by a total computable function if and only if  $n \in A$ , as wanted.

#### Solution 4.

A c.e. set is computable if and only its complement is c.e. Thus

$$\mathbf{Comp} = \{e \mid \exists i \forall n \forall t \exists t' > t (n \notin W_e[t'] \Leftrightarrow n \in W_i[t'])\}$$

hence  $\mathbf{Comp}$  is  $\Sigma_3^0$ .

To see that it is many-one complete, it suffices to analyze the construction of exercise 3. Given a  $\Sigma_3^0$  set  $A$ , exercise 3 shows how to, given  $n$  as a parameter, construct a partial computable function  $f$  such that

- If  $n \in A$ ,  $f$  has co-finite domain, hence its domain is computable
- If  $n \notin A$ ,  $f$  cannot be extended by any total computable function, hence by exercise 3 (b), its domain is not computable

Therefore, calling  $j(n)$  the index of the function  $f$ , we have  $n \in A \Leftrightarrow j(n) \in \mathbf{Comp}$ . Therefore any  $\Sigma_3^0$  set  $A$  is many-one reducible to  $\mathbf{Comp}$ .

