

## thy.1 Computably Enumerable Sets are Closed under Union and Intersection

cmp:thy:clo:  
sec

The following theorem gives some closure properties on the set of computably enumerable sets.

**Theorem thy.1.** *Suppose  $A$  and  $B$  are computably enumerable. Then so are  $A \cap B$  and  $A \cup B$ .*

*Proof.* ?? allows us to use various characterizations of the computably enumerable sets. By way of illustration, we will provide a few different proofs.

For the first proof, suppose  $A$  is enumerated by a computable function  $f$ , and  $B$  is enumerated by a computable function  $g$ . Let

$$\begin{aligned} h(x) &= \mu y (f(y) = x \vee g(y) = x) \text{ and} \\ j(x) &= \mu y (f((y)_0) = x \wedge g((y)_1) = x). \end{aligned}$$

Then  $A \cup B$  is the domain of  $h$ , and  $A \cap B$  is the domain of  $j$ .

Here is what is going on, in computational terms: given procedures that enumerate  $A$  and  $B$ , we can semi-decide if an element  $x$  is in  $A \cup B$  by looking for  $x$  in either enumeration; and we can semi-decide if an element  $x$  is in  $A \cap B$  by looking for  $x$  in both enumerations at the same time. explanation

For the second proof, suppose again that  $A$  is enumerated by  $f$  and  $B$  is enumerated by  $g$ . Let

$$k(x) = \begin{cases} f(x/2) & \text{if } x \text{ is even} \\ g((x-1)/2) & \text{if } x \text{ is odd.} \end{cases}$$

Then  $k$  enumerates  $A \cup B$ ; the idea is that  $k$  just alternates between the enumerations offered by  $f$  and  $g$ . Enumerating  $A \cap B$  is trickier. If  $A \cap B$  is empty, it is trivially computably enumerable. Otherwise, let  $c$  be any element of  $A \cap B$ , and define  $l$  by

$$l(x) = \begin{cases} f((x)_0) & \text{if } f((x)_0) = g((x)_1) \\ c & \text{otherwise.} \end{cases}$$

In computational terms,  $l$  runs through pairs of elements in the enumerations of  $f$  and  $g$ , and outputs every match it finds; otherwise, it just stalls by outputting  $c$ .

For the last proof, suppose  $A$  is the *domain* of the partial function  $m(x)$  and  $B$  is the domain of the partial function  $n(x)$ . Then  $A \cap B$  is the domain of the partial function  $m(x) + n(x)$ .

In computational terms, if  $A$  is the set of values for which  $m$  halts and  $B$  is the set of values for which  $n$  halts,  $A \cap B$  is the set of values for which both procedures halt. explanation

Expressing  $A \cup B$  as a set of halting values is more difficult, because one has to simulate  $m$  and  $n$  in parallel. Let  $d$  be an index for  $m$  and let  $e$  be an

index for  $n$ ; in other words,  $m = \varphi_d$  and  $n = \varphi_e$ . Then  $A \cup B$  is the domain of the function

$$p(x) = \mu y (T(d, x, y) \vee T(e, x, y)).$$

**explanation** In computational terms, on input  $x$ ,  $p$  searches for either a halting computation for  $m$  or a halting computation for  $n$ , and halts if it finds either one.  $\square$

## Photo Credits

## Bibliography