HW2-Eren-Karahüseyinoğlu

Zadanie 1. Without referring to the theorem about recursion with history, prove primitive recursiveness of the following functions:

(a) $\begin{cases} f(0) & = 0, \\ f(1) & = 1, \\ f(n+2) & = f(n+1) + f(n); \end{cases}$ (b) $\begin{cases} g(0) & = 1, \\ g(n+1) & = \sum_{i=0}^{n} g(i)^n. \end{cases}$

(a) Fibbonaci Sequence

$$f(0) = 0$$
$$f(1) = 1$$

$$f(n+1) = c(n+1) =< f(n), f(n-1) >$$

Now observe that if we look more into how fibonacci function works, we get:

$$\begin{gathered} f(n+1) = f(n) + f(n-1) \\ = \big(f(n-1) + f(n-2)\big) + f(n-1) \\ = \Big(\big(f(n-2) + f(n-3)\big) + f(n-2)\Big) + \big(f(n-2) + f(n-3)\big) \\ = \dots \end{gathered}$$

Here is the example of this:

$$\begin{gathered} f(4) = \dots \\ = \Bigg(\bigg(f(1) + f(0)\bigg) + f(1)\Bigg) + \bigg(f(1) + f(0)\bigg) \\ = ((1+0) + 1) + (1+0) = 3 \end{gathered}$$

It continues like this and it gives us the idea:
Idea is that encoding the pair of elements (previous two elements in this case) as a new element.

Now define the function c as:

For the base:

$$c(1) =< 1, 0 >$$

where $c_1(1) = 1$ and $c_2(1) = 0$. (Simply Projection) (Also may be called decoding function)

$$
\begin{gathered}
c(n+1) =< c(n), c(n-1) > \\
=< c_1(n) + c_2(n), c_1(n) > \\
= \dots
\end{gathered}
$$

All it does is enumerating the pairs of elements until it gets the basis elements.

Hence we can construct the function f by the function c. Since primitive recursiveness can be proven for the function c, we are done.

(b) For the case: Arbitrary number of elements

The difference from the previous question is that we need to record all of the previous elements to get the next element.

$$
\begin{gathered}
g(0) = 1 \\
g(n+1) = g(0)^n + g(1)^n + \dots + g(n)^n
\end{gathered}
$$

Define a function h such that: (taking n-tuple elements)

$$
\begin{gathered}
h(n+1) =< h(0), \dots, h(n) > \\
= h(0)^n + \dots + h(n)^n
\end{gathered}
$$

Where, for the base case ($h(0) = h(1) = 1$):

$$
\begin{gathered}
h(0) = 1 \\
h(1) =< h(0) >= h(0)^0 = 1 \\
h(2) =< h(0), h(1) >= h(0)^1 + h(1)^1 = 2
\end{gathered}
$$

where $h_0(2) = h(0) = 1$ and $h_1(2) = h(1) = 1$.

$$
\begin{gathered}
h(n+1) =< h(0), \dots, h(n) > \\
=< h_0(n), h_1(n), \dots, h_n(n) > \\
= h_0(n)^n + \dots + h_n(n)^n
\end{gathered}
$$

All we use here summation, *n th power and projection functions.*

As an example:

$$g(4) = g(0)^3 + g(1)^3 + g(2)^3 + g(3)^3$$
$$= g(0)^3 + \left(g(0)^0\right)^3 + \left(g(0)^1 + g(1)^1\right)^3 + \left(g(0)^2 + g(1)^2 + g(2)^2\right)^3$$
$$= 1^3 + \left(1^0\right)^3 + \left(1^1 + 1^1\right)^3 + \left(1^2 + 1^2 + 2^2\right)^3$$

Now, let us use the function h that we defined.

$$h(4) =< h(0), h(1), h(2), h(3) >$$
$$= h_0(3)^3 + h_1(3)^3 + h_2(3)^3 + h_3(3)^3$$

> Zadanie 2. Let $g : \mathbb{N} \to \mathbb{N}$ and $h_1, h_2 : \mathbb{N}^3 \to \mathbb{N}$ be primitive recursive. Prove that the function $f : \mathbb{N}^2 \to \mathbb{N}$ defined as follows:
>
> $$f(n,m) = \begin{cases} g(m) & if \quad n = 0, \\ h_1(n, m, f(n-2, m)) \quad if & n > 0 \; and \; 2|n, \\ h_2(n, m, f(2n, m)) & otherwise \end{cases}$$
>
> is also primitive recursive.

Here f is defined by course-of-values recursion by using primitive recursive function h_1 and h_2.

Now define t_1 and t_2 functions where $\pi(n) = n - 2$ and $\sigma(n) = 2n$:

$$t_1(n,m) = (n, m, f(\pi(n), m))$$
$$t_2(n,m) = (n, m, f(\sigma(n), m))$$

Observe that t_1 and t_2 functions are primitive recursive:

Let F be the list of functions such that $F(n,m) =< f(n,0), \ldots, f(n,m) >$ defined as encoded list. Observe that:

Now analyzing $F(n,m)$ we get:

$$F(0,m) =< f(0,m) >= g(m)$$

$$F(n+1,m) =< f(n+1,0), \ldots, f(n+1,m) >$$

where $f(n+1,m) = h_1(t_1(n+1,m))$ or $f(n+1,m) = h_2(t_1(n+1,m))$.

Now we need to observe that $t_1(n+1,m) = (n+1, m, f(\pi(n+1), m))$ where $\pi(n+1) < n$. Hence $f(\pi(n+1), m)$ is a component of $F(n+1,m)$.

Also observe that $t_2(n+1,m) = (n+1,m,f(\sigma(n+1),m))$ where $\sigma(n+1) \leq 2n+2$ but since we know $2|2n+2$ which means:
$f(2n+2,m)$ can be expressed by the component of $F(n+1,m)$ since $h_1(n,m,f(n-2,m))$

We need to define a new function G :

$$G(n,0,F(n,0)) =< G_0(n,0,F(n,0)) >$$

Assume that $G(n,m,F(n,m))$ is primitive recursive, then:

$$G(n,m+1,F(n,m+1)) = add(G(m,n,F(n,m)), G_0(n,m+1,F(n,m+1)))$$

where add is standard sequence operations which is primitive recursive and adds an element to the list.

****

> Zadanie 7. Provide a construction of a function f : N → N that is not primitive recursive.
> Hint. Use Cantor's diagonal argument.

We are going to define a function $diag$ by using Cantor's diagonal arguement.

Let us assume that following functions are primitive recursive:

$$\begin{array}{l} R_1(x) = a_{1,1}, a_{1,2}, a_{1,3} \dots \\ R_2(x) = a_{2,1}, a_{2,2}, a_{2,3} \dots \\ R_3(x) = a_{3,1}, a_{3,2}, a_{3,3} \dots \\ \vdots \qquad\qquad\qquad\qquad \ddots \end{array}$$

Now define the $diag$ function below:

$$diag(x) = d_1, d_2, \dots$$

where $d_i = a_{i,i} + 1$.

Observe that each R_i functions takes number of n inputs.
Claim that this $diag$ function is not primitive recursive. If this function were primitive recursive, it should have been appeared in the list above.

Let us say that this function is in the list, then it leads us to a contradiction that for some i, we have $a_{i,i} = d_i$ but it is imppossible by our definition of $diag$ function. Hence $diag$ is not primitive recursive.