4.3. Inductive definitions

Inductive definitions are increasingly being renamed to "recursive definitions" in the modern literature, thus using "recursive" for *definitions*, and "induction" for *proofs*. I will not go out of my way to use this dichotomy of nomenclature.

4.3.1 Example.

$$a^0 = 1$$
$$a^{n+1} = a \cdot a^n$$

is an example of an inductive (recursive) definition of the non-negative integer powers of a non zero number a. □

4.3.2 Example. Another example is the Fibonacci sequence,[†] given by

$$F_0 = 0$$
$$F_1 = 1$$
and for $n \geq 1$
$$F_{n+1} = F_n + F_{n-1}$$

Unlike the function (sequence) $a^0, a^1, a^2, a^3, \ldots$, for which we only need the value at n to compute the value at $n+1$, the Fibonacci function needs two previous values, at $n-1$ and at n, to compute the value at $n+1$. □

This section looks at inductive/recursive definitions in general, but for functions whose left field is $\mathbb{N}$ or $\mathbb{N}^{n+1}$ for some fixed n.

4.3.3 Definition. We consider in this section a general recursive definition of a function $G : \mathbb{N}^{n+1} \to A$, for a given $n \geq 0$ and set A.

This definition has the form (1) below.

Two total functions are *given*.

1. $H : \mathbb{N}^n \to A$, where A is some set. The typical *call* to H looks like $H(\mathbf{b})$ where $\mathbf{b} \in \mathbb{N}^n$. If $n = 0$, then we do *not* have any arguments for H. In this case H is just a *constant* (i.e., a fixed element of A).
2. $K : \mathbb{N}^{n+1} \times 2^A \to A$. The typical *call* to K looks like $K(m, \mathbf{b}, z)$ where $m \in \mathbb{N}$, $\mathbf{b} \in \mathbb{N}^n$ and z is a subset of A. If $n = 0$ then we do *not* have the argument $\mathbf{b}$.

   We will explore below whether the following definition (1) indeed yields a *function* $G : \mathbb{N}^{n+1} \to A$ of arguments a and $\mathbf{b}$ where $a \in \mathbb{N}$ and $\mathbf{b} \in \mathbb{N}^n$. If $n = 0$, then we do *not* have the argument $\mathbf{b}$, rather we will have just one argument in G: $a \in \mathbb{N}$.

[†] The "sequence" $F_0, F_0, F_0, \ldots$ is, of course, a total function from $F : \mathbb{N} \to \mathbb{N}$.

$$
\begin{aligned}
G(a, \mathbf{b}) &= H(\mathbf{b}) \\
G(a+1, \mathbf{b}) &= K\Big(a, \mathbf{b}, \Big\{G(0, \mathbf{b}), G(1, \mathbf{b}), \ldots, G(a, \mathbf{b})\Big\}\Big)
\end{aligned} \tag{1}
$$

□

4.3.4 Remark. The notation of the set-argument

$$\Big\{G(0, \mathbf{b}), G(1, \mathbf{b}), \ldots, G(a, \mathbf{b})\Big\} \tag{2}$$

in (1) above is *way less* informative than the notation implies! Its members —listed again in (2)— can be put in *any* order and *there are no markings on any of these members of* A that will reveal the 1st argument of G (the position of the call $G(i, \mathbf{b})$ in the sequence as presented in (2)). So we should not read (2) as if it conveys position!

Pause. Well, why not instead of using a *set*-argument write instead

$$K\Big(a, \mathbf{b}, G(0, \mathbf{b}), G(1, \mathbf{b}), \ldots, G(a, \mathbf{b})\Big)$$

that is, have each call to $G(i, \mathbf{b})$ explicitly "coded" in the function K? Because I cannot have a variable number of arguments!◀

This is *no problem in practise*. In any specific application of the **definition form** (1) the structure of K can be chosen/built so that it will "know and choose" what recursive calls it needs to make —in which order and for which arguments— to compute $G(a+1, \mathbf{b})$.

For example, the specific use of principle (1) to the Fibonacci function definition 4.3.2 has chosen that to compute F_{n+1} it will always call just F_n and F_{n-1} from the entire "history at input n" —namely, $\{F_0, F_1, F_2, \ldots, F_n\}$— and then return the sum of the call results.

So the notation (1) (via (2)) simply conveys —for the benefit of our two theorems coming up below— that *in general* an inductive definition (1) might call recursively as many as all the $G(i, \mathbf{b})$ in (2) to compute $G(a+1, \mathbf{b})$.

BTW, there are complicated inductive definitions such that the recursive calls are not always at fixed (argument-)positions to the left of "$a+1$", unlike the Fibonacci recursive definition that computes F_{n+1}, for any $n \geq 1$, by always calling the function recursively with arguments *at precisely the numbers before* $n+1$. These complicated cases will choose which $G(i, \mathbf{b})$ from among the history (2) to call, depending on the value of $a+1$ □

4.3.5 Lemma. *Let* $n \geq 1$. *If we define the order* $\prec$ *on* $\mathbb{N}^{n+1}$ *by* $(a, \mathbf{b}) \prec (a', \mathbf{b}')$ *iff* $a < a'$ *and* $\mathbf{b} = \mathbf{b}'$, *then* $\prec$ *is an order that has MC on* $\mathbb{N}^{n+1}$.

Proof.

1. $\prec$ is an order:
    - Indeed, if $(a, \mathbf{b}) \prec (a, \mathbf{b})$, then $a < a$ which is absurd.

- If $(a, \mathbf{b}) \prec (a', \mathbf{b}') \prec (a'', \mathbf{b}'')$, then $\mathbf{b} = \mathbf{b}' = \mathbf{b}''$ and $a < a' < a''$. Thus $a < a''$ and hence $(a, \mathbf{b}) \prec (a'', \mathbf{b}'')$.

2. $\prec$ has MC: So let $\emptyset \neq A \subseteq \mathbb{N}^{n+1}$. Let a be $<$-minimum in $S = \{x : (\exists \mathbf{b})(x, \mathbf{b}) \in A\} \subseteq \mathbb{N}$.

   Pause. Why is $S \neq \emptyset$?◀

   Let $\mathbf{c}$ be such that $(a, \mathbf{c}) \in A$. This $(a, \mathbf{c})$ is $\prec$-minimal in A. Otherwise for some d, $A \ni (d, \mathbf{c}) \prec (a, \mathbf{c})$. Hence $d < a$, but this is a contradiction since $d \in S$ (why?). □

The minimal elements of $\prec$ are of the form $(0, \mathbf{b}), (0, \mathbf{b}'), (0, \mathbf{b}''), \ldots$, which are not comparable if they have distinct "**b**-parts". Thus they are infinitely many.

4.3.6 Lemma. *Let $(Y, <)$ be a POset with MC —where I use "$<$" generically, not as the one on $\mathbb{N}$.*

Then, for any subset $\emptyset \neq B$ of Y, $(B, <)$ is a POset with MC.

Proof. We show two things:

1. $(B, <)$ is a POset.

   $<$ is irreflexive on Y, hence it is trivially so on any subset of Y. Transitivity too is inherited from that of $<$ on Y, since if x, y, z are in B and we have $x < y < z$, then x, y, z are in Y and we still have $x < y < z$. Hence $x < z$ is true.

2. Let $\emptyset \neq S \subseteq B$. Now S —viewed as a subset of Y— has a $<$-*minimal* member m. We cannot have $x < m$ with $x \in S$ in $(B, <)$ since then we have $x < m$ with $x \in S$ in $(Y, <)$. □

4.3.7 Theorem. *If there is a function $G : \mathbb{N}^{n+1} \to A$ satisfying (1) of 4.3.3, then it is unique.*

Proof. Suppose we have two such functions, G and G' that satisfy (1) for **given** H and K. If G and G' differ, then there is an argument $(a, \mathbf{b})$ such that $G(a, \mathbf{b}) \neq G'(a, \mathbf{b})$ then there is —by Lemma 4.3.5— a $\prec$-*minimal* such argument, say, $(m, \mathbf{c})$, in the set $T = \{(a, \mathbf{b}) : G(a, \mathbf{b}) \neq G'(a, \mathbf{b})\}$. So

$$G(m, \mathbf{c}) \neq G'(m, \mathbf{c}) \tag{$*$}$$

Now, $(m, \mathbf{c})$ is *not* $\prec$-minimal in $\mathbb{N}^{n+1}$ since on such inputs we have $G(0, \mathbf{d}) = H(\mathbf{d}) = G'(0, \mathbf{d})$. Thus, in particular, $m > 0$.

But then, by (1) of 4.3.3, we compute each of $G(m, \mathbf{c})$ and $G'(m, \mathbf{c})$ by the *second equation* as

$$K\Big(m - 1, \mathbf{c}, \{G(0, \mathbf{c}), G(1, \mathbf{c}), \ldots, G(m - 1, \mathbf{c})\}\Big)$$

since minimality of $(m, \mathbf{c})$ in the set T entails

$$G(i, \mathbf{c}) = G'(i, \mathbf{c}), \text{ for } i = 0, 1, \ldots m - 1$$

Since K is single-valued (function!) we have $G(m, \mathbf{c}) = G'(m, \mathbf{c})$, contradicting $(*)$. Thus $T = \emptyset$ and therefore $G(a, \mathbf{b}) = G'(a, \mathbf{b})$, for all $(a, \mathbf{b}) \in \mathbb{N}^{n+1}$. For short, the functions G and G' are the same. □

4.3.8 Theorem. *There is* *a function $G : \mathbb{N}^{n+1} \to A$ satisfying (1) of 4.3.3.*

Proof. The idea is simple: Build the function by stages as an infinite set of building blocks. Each block is a *restriction* of G —that is, a partial table for G— so that the domain of the restriction is an "*initial segment*" of $\mathbb{N}^{n+1}$ determined by some point ("point" is synonymous to "element") $(m, \mathbf{b})$. Thus the "general" segment is the set

$$S_{(m,\mathbf{b})} \stackrel{Def}{=} \{(a, \mathbf{b}) : (a, \mathbf{b}) \prec (m, \mathbf{b})\} \cup \{(m, \mathbf{b})\} \qquad (\dagger)$$

The notation "$S_{(m,\mathbf{b})}$" reflects "S" for *segment*, subscripted with the defining point $(m, \mathbf{b})$. Once you have *all* the building blocks, you put them together to get the G you want.

Let us call $G_{(m,\mathbf{b})}$ *the* function (if it exists) from $S_{(m,\mathbf{b})} \to A$ that satisfies (1) of 4.3.3 if we replace the G there by $G_{(m,\mathbf{b})}$ everywhere.

Why am I emphasising "the"? Because $S_{(m,\mathbf{b})}$ inherits MC from N^n. Cf. 4.3.6. And then 4.3.7 applies to $G_{(m,\mathbf{b})} : S_{(m,\mathbf{b})} \to A$ as the proof of 4.3.7 applies unchanged (just change $\mathbb{N}^{n+1}$ and G to $S_{(m,\mathbf{b})}$ and $G_{(m,\mathbf{b})}$ respectively; all else is the same in the proof).

We have one more **important** (for this proof) observation related to uniqueness: If $\boxed{(x, \mathbf{b}) \prec (y, \mathbf{b}), \text{ then } G_{(x,\mathbf{b})}(u, \mathbf{b}) = G_{(y,\mathbf{b})}(u, \mathbf{b}), \text{ for all } u \le x}$.[†]

Indeed, if $G_{(x,\mathbf{b})}$ and $G_{(y,\mathbf{b})}$ exist, then they both satisfy (1) of 4.3.3 on the subset $S_{(x,\mathbf{b})}$ of $S_{(y,\mathbf{b})}$.

Our next task is simply to show that for each $(m, \mathbf{b}) \in \mathbb{N}^{n+1}$,

the function $G_{(m,\mathbf{b})} : S_{(m,\mathbf{b})} \to A$ that satisfies (1) in 4.3.3 *exists* $\qquad (\ddagger)$

where we changed $\mathbb{N}^{n+1}$ and G into $S_{(m,\mathbf{b})}$ and $G_{(m,\mathbf{b})}$ respectively.

We do so *constructively* —that is, show how each $G_{(m,\mathbf{b})} : S_{(m,\mathbf{b})} \to A$ is *built*— by CVI on the variable $(m, \mathbf{b})$ along the order $\prec$ over $\mathbb{N}^{n+1}$.

1. *Basis*: For *any* minimal $(0, \mathbf{b})$,[‡] we have $S_{(0,\mathbf{b})} = \{(0, \mathbf{b})\}$. Thus, using the first equation of (1) in 4.3.3, we set

$$G_{(0,\mathbf{b})} = \Big\{\big((0, \mathbf{b}), H(\mathbf{b})\big)\Big\}^{\S}$$

[†] Here "$\le$" is, of course, the "less-than-or-equal" on $\mathbb{N}$.

[‡] We remarked that the $(0, \mathbf{b})$ for various $\mathbf{b} \in \mathbb{N}^n$ *are* the $\prec$-minimal points in $\mathbb{N}^{n+1}$.

[§] We still remember that a function is a set of pairs! This *one* has just one pair.

2. *I.H.* Assume that for all $(x, \mathbf{b}) \prec (m, \mathbf{b})$[†] we have built $G_{(x,\mathbf{b})} : S_{(x,\mathbf{b})} \to A$ all of which satisfy (the two equations of) (1) of 4.3.3.

   In view of the boxed statement above, $G_{(m,\mathbf{b})}$ coincides with each $G_{(x,\mathbf{b})}$ —for $(x, \mathbf{b}) \prec (m, \mathbf{b})$— on the latter's domain. Thus I need only add one input/output pair to $\bigcup_{(x,\mathbf{b})\prec(m,\mathbf{b})} G_{(x,\mathbf{b})} = G_{(m-1,\mathbf{b})}$

   Why is this last "=" correct?

   at input $(m, \mathbf{b})$ to obtain $G(m, \mathbf{b})$.

   To do so I simply use (1) of 4.3.3, second equation. The I/O pair added to obtain $G_{(m,\mathbf{b})}$ is

$$\Big((m-1, \mathbf{b}),\ K\big(m-1, \mathbf{b}, \{G_{(m-1,\mathbf{b})}(0, \mathbf{b}), \ldots, G_{(m-1,\mathbf{b})}(m-1, \mathbf{b})\}\big)\Big)$$

   It is clear that on *any* input $(u, \mathbf{b})$, whether the just *constructed relation* $G_{(m,\mathbf{b})}$ "thinks" that it is $G_{(x,\mathbf{b})}$ or $G_{(y,\mathbf{b})}$ it will give *the same output* due the boxed statement above. Thus, the relation $G_{(x,\mathbf{b})}$ is a *function*.

It is now time to put all the $G_{(x,\mathbf{b})}$ together to form $G : \mathbb{N}^{n+1} \to A$. Just define G by

$$G \stackrel{Def}{=} \bigcup_{(x,\mathbf{b})\in\mathbb{N}^{n+1}} G_{(x,\mathbf{b})} \tag{$*$}$$

Observe regarding G:

1. As a *relation* it is total on the left field $\mathbb{N}^{n+1}$ because it is *defined* on the arbitrary $(x, \mathbf{b}) \in \mathbb{N}^{n+1}$ since $G_{(x,\mathbf{b})} : S_{(x,\mathbf{b})} \to A$ is.

2. $\mathrm{ran}(G) \subseteq A$. Because it is so for each $G_{(x,\mathbf{b})} : S_{(x,\mathbf{b})} \to A$.

3. G is single-valued, hence a *function* from $\mathbb{N}^{n+1}$ to A, since the value $G(u, \mathbf{b})$ does not depend on which $G_{(x,\mathbf{b})} : S_{(x,\mathbf{b})\to A}$ we used to obtain it as $G_{(x,\mathbf{b})}(u, \mathbf{b})$ (by boxed statement above).

   Finally,

4. G satisfies (1) of 4.3.3 since by $(*)$, for any $(x, \mathbf{b}) \in \mathbb{N}^{n+1}$, $G(x, \mathbf{b}) = G_{(x,\mathbf{b})}(x, \mathbf{b})$, and $G_{(x,\mathbf{b})}(x, \mathbf{b})$ is constructed to obey the two equations of (1) of 4.3.3, for all $x \geq 0$ and $\mathbf{b} \in \mathbb{N}^n$. □

Let us see some examples:

4.3.9 Example. We know that 2^n means

$$\overbrace{2 \times 2 \times 2 \times \ldots \times 2}^{n\ 2s}$$

[†]Recall that for $\mathbf{b} \neq \mathbf{c}$, $(x, \mathbf{b})$ and $(y, \mathbf{c})$ are not comparable.

But "...", or "etc.", is *not* MATH! That is why we gave at the outset of this section the definition 4.3.1.

Applied to the case $a = 2$ we have

$$\begin{aligned} 2^0 &= 1 \\ 2^{n+1} &= 2 \times 2^n \end{aligned} \tag{1}$$

We know from 4.3.8 and 4.3.7 that both (1) above and the definition in 4.3.1 define a unique function, each satisfying its defining equations.

For the function that for each n outputs 2^n we can give an alternative definition that uses "+" rather than "×":

$$\begin{aligned} 2^0 &= 1 \\ 2^{n+1} &= 2^n + 2^n \end{aligned}$$

□

4.3.10 Example. Let $f : \mathbb{N}^{n+1} \to \mathbb{N}$ be given. How can I define $\sum_{i=0}^{n} f(i, \mathbf{b})$ —for any $\mathbf{b} \in \mathbb{N}^n$— other than by the sloppy

$$f(0, \mathbf{b}) + f(1, \mathbf{b}) + f(2, \mathbf{b}) + \ldots + f(i, \mathbf{b}) + \ldots + f(n, \mathbf{b})?$$

By induction/recursion, of course:

$$\begin{aligned} \textstyle\sum_{i=0}^{0} &= f(0, \mathbf{b}) \\ \textstyle\sum_{i=0}^{n+1} &= \Big(\textstyle\sum_{i=0}^{n} f(i, \mathbf{b}) \Big) + f(n+1, \mathbf{b}) \end{aligned} \tag{1}$$

□

4.3.11 Example. Let $f : \mathbb{N}^{n+1} \to \mathbb{N}$ be given. How can I define $\prod_{i=0}^{n} f(i, \mathbf{b})$ —for any $\mathbf{b} \in \mathbb{N}^n$— other than by the sloppy

$$f(0, \mathbf{b}) \times f(1, \mathbf{b}) \times f(2, \mathbf{b}) \times \ldots \times f(i, \mathbf{b}) \times \ldots \times f(n, \mathbf{b})?$$

By induction/recursion, of course:

$$\begin{aligned} \textstyle\prod_{i=0}^{0} &= f(0, \mathbf{b}) \\ \textstyle\prod_{i=0}^{n+1} &= \Big(\textstyle\prod_{i=0}^{n} f(i, \mathbf{b}) \Big) + f(n+1, \mathbf{b}) \end{aligned} \tag{2}$$

Again, by 4.3.8 and 4.3.7, each of (1) and (2) define a unique function, $\sum$ and $\prod$ that behaves as required. Really? For example, the first equation of (1) gives us the one-term sum, $f(0, \mathbf{b})$. It is correct. Assume (I.H. by simple induction on n) that the term $\sum_{i=0}^{n} f(i, \mathbf{b})$ correctly captures the sloppy

$$f(0, \mathbf{b}) + f(1, \mathbf{b}) + f(2, \mathbf{b}) + \ldots + f(i, \mathbf{b}) + \ldots + f(n, \mathbf{b})$$

that indicates the sum of the first $n + 1$ terms of the type $f(i, \mathbf{b})$ for $i = 0, 1, 2, \ldots, n$. But then, clearly the second equation of (1) correctly defines the sum of the first $n + 2$ terms of the above type, by adding $f(n + 1, \mathbf{b})$ to $\sum_{i=0}^{n} f(i, \mathbf{b})$. □

4.3.12 Example. Here is a function with huge output! Define $f : \mathbb{N} \to \mathbb{N}$ by

$$\begin{aligned} f(0) &= 1 \\ f(n+1) &= 2^{f(n)} \end{aligned} \tag{3}$$

What does $f(n)$ look like in sloppy notation? Well,

$$f(0) = 1, \quad f(1) = 2^{f(0)} = 2, \quad f(2) = 2^{f(1)} = 2^2, \quad f(3) = 2^{f(2)} = 2^{2^2}$$

Hmm! Is the guess that $f(n)$ is a ladder of n 2s? Yes! Let's verify by induction:

1. *Basis.* $f(0) = 1$. A ladder of zero 2s. Correct.
2. *I.H.* Fix n and assume that

$$f(n) = \left. 2^{2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}} \right\} n \text{ 2s}$$

   A ladder of n 2s.
3. I.S. Thus $f(n+1) = 2^{f(n)}$, so we put the ladder of n 2s of the I.H. as the exponent of 2 —forming a ladder of $n+1$ 2s— to obtain $f(n+1)$. Done! □

4.3.13 Example. (Fibonacci; a comment) This short example is to be clear, as in the case of induction proofs, that the "Basis" case is for minimal elements (compare with Exercise 4.2.13, case 5).

$$\begin{aligned} F_0 &= 0 \\ F_1 &= 1 \\ &\text{and for } n \ge 1 \\ F_{n+1} &= F_n + F_{n-1} \end{aligned}$$

In the above "$F_1 = 1$" is *NOT* a "Basis case" because 1 is *not* minimal in $\mathbb{N}$! ("$F_0 = 0$" *is* the Basis case, corresponding to the first equation in (1) of 4.3.3). So what is "$F_1 = 1$"? It is a boundary case of the second equation in the general Definition 4.3.3. This equation, in the Fibonacci case, can be rewritten as

$$F_{n+1} = \text{if } n = 0 \text{ then } 1 \\ \qquad \text{else } F_n + F_{n-1}$$

□