1 Introduction

Recursive definitions and inductive proofs are the twin poster children of theoretical computer science. Recursively defined sets and data structures are ubiquitous, and induction is the tool that allows them to be analyzed. Before embarking on the formalities let's consider a toy example.

Definition: A is the smallest subset of $\mathbb{N}$ having both the following properties:

- $0 \in A$
- If $x \in A$ then $2x \in A$ and $4x + 1 \in A$.

Problem: Prove that A is the set of natural numbers whose binary expansions do not contain consecutive 1s.

The first question to ask is: does the definition make sense? Can we be sure there is any such set A? Can we be sure there is a smallest one? The way in which the definition is formed should make it clear that the answer in each case is yes. The first part forces us to include the element 0. The second part gives us a recipe for adding new elements to A given some old ones. But the key thing is that the "new" elements are in some sense more complex (specifically, larger) than the old ones. So we get a sequence of larger and larger numbers that we're obligated to include in A, but provided we do include these *and no others*, then both conditions of the definition will be met.

The second question to ask is: how can we prove things about A? Again, the word "smallest" in the definition comes to our rescue. Suppose that we have some element of A – we want to show that it has no consecutive 1's in its binary expansion. If that element is 0 this is clear. If not, then it must be of the form $2x$ or $4x + 1$ for some simpler element $x \in A$. Supposing we already knew the result were true for x then we see that the binary expansion of the element we're interested in is obtained from that of x by appending 0 or 01. So, we can never create consecutive 1's. Conversely, if we have a number whose binary expansion contains no consecutive 1s and it is not 0 then it either ends with 0, or ends with 01, so it is either $2x$ or $4x + 1$ for some other number of this type, and so must belong to A.

The key point to justify in the argument above is: why can we assume the result we are trying to prove for all simpler (in this case smaller) numbers? Well, imagine that the result were false. Then either there would be a smallest number belonging to A which did have consecutive 1's, or there would be a smallest number not having consecutive 1's but not belonging to A. In both cases "the result" is true for all numbers smaller than this counterexample – and the argument then shows that the counterexample can't actually exist.

2 Recursive definitions

A *recursive definition* of some set, A, of objects consists of two explicit and one implicit parts:

- A *basis* or set of *atomic cases* which is a list of some objects that must belong to A;
- A *recursive step* which is a rule or collection of rules for producing new objects in A from existing ones (typically these rules are given as functions which, when applied to elements of A, and possibly some other basic objects, generate new elements of A);
- An implicit rule that A is the smallest set of elements satisfying the previous two conditions.

Example: recursive definition of $\mathbb{N}$.

First define a function s on sets as follows:

$$s(X) = X \cup \{X\}$$

That is, $s(X)$ is the set whose elements are the elements of X and the set X itself.

Now we can define $\mathbb{N}$ recursively:

- $\emptyset \in \mathbb{N}$
- If $n \in \mathbb{N}$ then $s(n) \in \mathbb{N}$.

Using 0 as a shorthand for the empty set:

$$\mathbb{N} = \{0, s(0), s(s(0)), s(s(s(0))), \dots\}$$

And we get the more 'normal' notation by just counting s occurrences.

More usually we might start with some predefined set (like $\mathbb{N}$) and recursively define a subset of it. For example, how might we define the set of EVEN numbers? One way would be to say "numbers which are a multiple of 2", but this is not constructive (and relies on us knowing more about arithmetic). We could do it recursively:

- $0 \in$ EVEN
- If $n \in$ EVEN then $s(s(n)) \in$ EVEN

3 Inductive proofs

Having formed recursively defined sets we will want to prove things about them. The rigidity of the rules that grant membership in such a set mean that we have a tailor-made proof principle at hand called *induction*. The basic set up is this: we want to prove that every element a of a recursively designed set A has some property P. It will be both necessary and sufficient to show that:

- All the elements specified by the base case of the recursive definition have property P
- If the constructive rules (recursive steps) are applied to elements having property P then the resulting elements also have property P.

Then, the implicit "smallest" condition means that no element is ever added to A that does not have property P.

Consider the set of binary trees with natural numbers as keys defined as follows:

- A natural number is a binary tree (called a leaf)

- If T_L and T_R are binary trees and n is a natural number then the triple $T = (n, T_L, T_R)$ is a binary tree.

Define the number of leaves of a binary tree to be 1 if it is a leaf, and the sum of the number of leaves of T_L and T_R if the second case applies. Define the number of internal nodes of a binary tree to be 0 if it is a leaf, and 1 plus the sum of the number of internal nodes of T_L and T_R in the latter case.

Problem: Prove that in any binary tree, the number of internal nodes is one less than the number of leaves.

This becomes quite simple when we use the inductive method. It's obviously true in the base case, since there we have one leaf, and zero internal nodes. If it's true of T_L and T_R then we just compute:

$$\begin{aligned} \mathit{internal}(T) &= 1 + \mathit{internal}(T_L) + \mathit{internal}(T_R) \\ &= 1 + \mathit{leaf}(T_L) - 1 + \mathit{leaf}(T_R) - 1 \\ &= \mathit{leaf}(T_L) + \mathit{leaf}(T_R) - 1 \\ &= \mathit{leaf}(T) - 1. \end{aligned}$$

The recursive definition of $\mathbb{N}$ allows for a particular form of induction called *mathematical induction* (which is sometimes the only kind that people learn about). We can express it in either a weak or strong form. The weak form is:

> If P is a property of natural numbers such that $P(0)$ is true, and whenever $P(k)$ is true then $P(k+1)$ is true, then $P(n)$ is true for all $n \in \mathbb{N}$.

The strong form is:

> If P is a property of natural numbers such that $P(0)$ is true, and whenever $P(j)$ is true for all $j < k$ then $P(k)$ is true, then $P(n)$ is true for all $n \in \mathbb{N}$.

To see that there really is no difference, think of climbing a ladder. The weak form says "if you can always get to the next step from the previous one, you can go as high as you

like", while the strong form says "if, having gotten this far, you can always get one step further, then you can go as high as you like". It only appears to be stronger since in the chain of argument leading up to determining that $P(k)$ is true (for the weak form) we will have implicitly verified it for all smaller cases already.

The standard types of examples for mathematical induction are to verify formulas for e.g. sums. For instance:

$$1 + 4 + 9 + \cdots + n^2 = \frac{n(n+1)(2n+1)}{6}$$

This is true for $n = 0$ (both sides are 0). Suppose it to be true for $n = k$ and compute:

$$\begin{aligned} 1 + 4 + 9 + \cdots + (k+1)^2 &= 1 + 4 + 9 + \cdots + k^2 + (k+1)^2 \\ &= \frac{k(k+1)(2k+1)}{6} + (k+1)^2 \\ &= \frac{(k+1)(2k^2 + k + 6(k+1))}{6} \\ &= \frac{(k+1)(k+2)(2k+3)}{6} \end{aligned}$$

which is what we want since $2k + 3 = 2(k+1) + 1$.

As a quick example of the strong version let's consider the problem: prove that every natural number greater than 1 is either prime (i.e. has no proper factors), or is a product of primes.

Let a natural number $n > 1$ be given, and suppose the result is true for all smaller natural numbers. If n is prime there is nothing to prove. So suppose that n has a proper factor, a, say $n = ab$. But, by induction, both a and b are either primes or products of primes, and hence so is n.

To be completely honest here we're glossing over the fact that the property P we're using is "$n \leq 1$ or n is either prime or a product of primes" – such inessential modifications are quite common in either form – e.g. in the weak form we might not start from 0 but from 3 and then the result is "for all $n \geq 3$, $P(n)$ holds.

4 Exercises

Recursive definitions

1. Give a recursive definition of the set $P = \{1, 2, 4, 8, 16, \ldots\}$ of powers of two within $\mathbb{N}$. (You may assume the operation $+$ on $\mathbb{N}$ has already been defined.)
2. Give a recursive definition of the subset, Eq, of $\mathbb{N} \times \mathbb{N}$ representing the relation *is equal to* using the successor function $s(n) = n + 1$.
3. Give a recursive definition of "linked list of natural numbers" (hint: the base case should be equivalent to `nil`, and the construction should use a pair).
4. (Harder) Give a recursive definition of the set of *finite* subsets of $\mathbb{N}$, using the successor function s and union as the operators.

Inductive proofs

1. Prove that $2 + 5 + 8 + \cdots + (3n - 1) = n(3n + 1)/2$ for all $n > 0$.
2. Prove that $1 + 2 + 2^2 + \cdots + 2^n = 2^{n+1} - 1$ for all $n \geq 0$.
3. Suppose that two algorithms do the same thing, but the first requires $f(n) = 4n + 1$ steps, while the second requires $g(n) = n^2$ steps. For small values of n, the second algorithm is better, but it seems clear that when n is "big enough" we should prefer the first. Make this precise by proving that $f(n) < g(n)$ for all $n \geq 5$.
4. Consider the following recursively defined function $f : \mathbb{N} \to \mathbb{N}$.

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

   Compute the first few values of f, form a conjecture for a more "natural" description of f, and prove that your conjecture is correct.

5. Let a and b be two symbols. Recursively define a set of strings (sequences), D as follows:

   Basis The empty string is in D.

   Recursive step If a string s is in D, then so is the string asb. If strings s and t are in D then so is the string st.

   (a) List all the strings in D consisting of 6 or fewer symbols.

   (b) Prove by induction that every string in D has even length.

   (c) Prove by induction that every string in D has the same number of a's as b's.

   (d) Prove by induction that in any prefix of a string in D there are at least as many a's as b's.

6. Consider the following piece of pseudocode defining a function $\mathsf{foo}(x, y)$, where we assume $x, y \in \mathbb{N}$ (the division operator is the usual e.g., Java, integer division, and % is the remainder operation). To maintain some mathematical integrity we use $=$ in the mathematical sense (i.e. of an equality test) and use $\leftarrow$ to denote assignment.

   if $x = 0$ **then**
   `return` 0
   end if
   $s \leftarrow \mathsf{foo}(x/2, 2y)$
   if $x\%2 \neq 0$ **then**
   $s \leftarrow s + y$
   end if
   `return` s

   Compute a table of values for foo, form a conjecture about its more natural definition, and prove that conjecture by induction.