Lecture Notes:
Discrete Mathematics for Computer Science

Vladimir Lifschitz
University of Texas at Austin

Part 4. Induction and Recursion

Proofs by Induction

Induction is a useful proof method in mathematics and computer science. When we want to prove by induction that some statement containing a variable n is true for all nonnegative values of n, we do two things. First we prove the statement when $n = 0$; this part of the proof is called the *basis*. Then we prove the statement for $n+1$ assuming that it is true for n; this part of the proof is called the *induction step*. (The assumption that the statement is true for n, which is used in the induction step, is called the *induction hypothesis*.)

Once we have completed both the basis and the induction step, we can conclude that the statement holds for all nonnegative values of n. Indeed, according to the basis, it holds for $n = 0$. From this fact, according to the induction step, we can conclude that it holds for $n = 1$. From this fact, according to the induction step, we can conclude that it holds for $n = 2$. And so on.

As an example, we will give yet another proof of the formula for triangular numbers.

Problem. Prove that for all nonnegative integers n

$$1 + 2 + \cdots + n = \frac{n(n+1)}{2}.$$

Solution. *Basis.* When $n = 0$, the formula turns into

$$0 = \frac{0(0+1)}{2},$$

which is correct. *Induction step.* Assume that

$$1 + 2 + \cdots + n = \frac{n(n+1)}{2}.$$

We need to derive from this assumption that

$$1 + 2 + \cdots + n + (n+1) = \frac{(n+1)(n+2)}{2}.$$

Using the induction hypothesis, we calculate:

$$\begin{aligned} 1+2+\cdots+n+(n+1) &= \frac{n(n+1)}{2}+(n+1) \\ &= \frac{n(n+1)+2(n+1)}{2} \\ &= \frac{(n+1)(n+2)}{2}. \end{aligned}$$

Proofs by induction can be symbolically represented by this inference rule:

$$\frac{P(0) \qquad \forall n(P(n) \rightarrow P(n+1))}{\forall n P(n)}.$$

Here n is a variable for nonnegative integers, and the expression $P(n)$ means that n has the property that we want to prove holds for n. The first premise represents the basis, and the second premise represents the induction step.

Two More Examples of Induction

Problem. Prove that for all nonnegative integers n, $2^n > n$.

Solution. *Basis.* When $n = 0$, the formula turns into $1 > 0$, which is correct. *Induction step.* Assume that $2^n > n$. We need to derive from this assumption that $2^{n+1} > n + 1$. This can be done as follows, using the induction hypothesis and then the fact that $2^n \geq 1$:

$$2^{n+1} = 2^n + 2^n > n + 2^n \geq n + 1.$$

Problem. Prove that for all nonnegative integers n, $n^3 - n$ is a multiple of 3.

Solution. *Basis.* When $n = 0$, we need to check that $0^3 - 0$ is a multiple of 3, which is correct. *Induction step.* Assume that $n^3 - n$ is a multiple of 3. We need to derive from this assumption that $(n + 1)^3 - (n + 1)$ is a multiple of 3. This expression can be rewritten as follows:

$$\begin{aligned} (n+1)^3-(n+1) = n^3+3n^2+3n+1-(n+1) &= n^3+3n^2+2n \\ &= (n^3-n)+3n^2+3n. \end{aligned}$$

Consider the three summands $n^3 - n$, $3n^2$, $3n$. By the induction hypothesis, the first of them is a multiple of 3. It is clear that the other two are multiples of 3 also. Consequently, the sum is a multiple of 3.

Starting with a Number Other than 0

We have used induction to prove statements about *nonnegative* integers. Statements about *positive* integers can be proved by induction in a similar way, except that the basis corresponds to $n = 1$; also, in the induction step we may assume that $n \geq 1$. Similarly, if we want to prove a statement about all integers beginning with 2 then the basis corresponds to $n = 2$, and so on.

Problem. Prove that for all integers n such that $n \geq 10$, $2^n > n + 1000$.

Solution. *Basis.* When $n = 10$, the formula turns into $1024 > 1010$, which is correct. *Induction step.* Assume that $2^n > n + 1000$ for an integer n such that $n \geq 10$. We need to derive that $2^{n+1} > n + 1001$. This can be done as follows:

$$2^{n+1} = 2^n + 2^n > n + 1000 + 2^n > n + 1000 + 1 = n + 1001.$$

Recursive Definitions

A *recursive definition* of a sequence of numbers expresses some members of that sequence in terms of its other members. For instance, here is a recursive definition of triangular numbers:

$$\begin{aligned} &T_0 = 0, \\ &T_{n+1} = T_n + n + 1. \end{aligned}$$

The first formula gives the first triangular number explicitly; the second formula shows how to calculate any other triangular number if we already know the previous triangular number.

There are two ways to find T_4 using this definition. One is to find first T_1, then T_2, then T_3, and then T_4:

$$\begin{aligned} T_1 &= T_0 + 1 = 0 + 1 = 1, \\ T_2 &= T_1 + 2 = 1 + 2 = 3, \\ T_3 &= T_2 + 3 = 3 + 3 = 6, \\ T_4 &= T_3 + 4 = 6 + 4 = 10. \end{aligned}$$

The other possibility is to form a chain of equalities that begins with T_4 and ends with a number:

$$\begin{aligned} T_4 &= T_3 + 4 \\ &= T_2 + 3 + 4 = T_2 + 7 \\ &= T_1 + 2 + 7 = T_1 + 9 \\ &= T_0 + 1 + 9 = T_0 + 10 \\ &= 0 + 10 = 10. \end{aligned}$$

This is an example of "lazy evaluation": we don't calculate the members of the sequence other than our goal T_4 until they are needed. The strategy used in the first calculation is "eager," or "strict."

If a sequence of numbers is defined using Sigma-notation then we can always rewrite its definition using recursion. For instance, the formula

$$X_n = \sum_{i=1}^{n} \frac{1}{i^2 + 1}$$

can be rewritten as

$$X_0 = 0,$$
$$X_{n+1} = X_n + \frac{1}{(n+1)^2+1}.$$

The sequence of factorials can be described by a recursive definition also:

$$0! = 1,$$
$$(n+1)! = n! \cdot (n+1).$$

Proving Properties of Recursively Defined Sequences

To prove properties of recursively defined sequences, we often use induction. Consider, for instance, the numbers $Y_0, Y_1, Y_2, \ldots$ defined by the formulas

$$Y_0 = 0,$$
$$Y_{n+1} = 2Y_n + n + 1.$$

We will prove by induction that $Y_n \geq 2^n$ whenever $n \geq 2$. *Basis:* $n = 2$. Since $Y_2 = 4$ and $2^2 = 4$, the inequality $Y_2 \geq 2^2$ holds. *Induction step.* Assume that $Y_n \geq 2^n$ for an integer n such that $n \geq 2$. We need to derive that $Y_{n+1} \geq 2^{n+1}$. This can be done as follows:

$$Y_{n+1} = 2Y_n + n + 1 \geq 2 \cdot 2^n + n + 1 = 2^{n+1} + n + 1 > 2^{n+1}.$$

Recursive Definitions in Case Notation

Recursive definitions can be written in "case notation" by showing which formula should be used for calculating the n-th member of the sequence depending on the value of n. For instance, the definition of triangular numbers, rewritten in case notation, will look like this:

$$T_n = \begin{cases} 0, & \text{if } n = 0, \\ T_{n-1} + n, & \text{otherwise.} \end{cases}$$

In all examples of recursive definitions so far, the larger n is, the more work is needed to calculate the n-th member of the sequence. The recursive definition of the numbers $M(0), M(1), M(2), \ldots$ shown below is different: it's easy to calculate $M(n)$ when n is large, and difficult when n is small.

$$M(n) = \begin{cases} n - 10, & \text{if } n > 100, \\ M(M(n+11)), & \text{otherwise.} \end{cases}$$