PROVING PROGRAMS CORRECT

ROBERT B. ANDERSON

PROVING PROGRAMS CORRECT

ROBERT B. ANDERSON

University of Houston

JOHN WILEY & SONS

New York Chichester Brisbane Toronto

Library of Congress Cataloging in Publication Data

Anderson, Robert Brockett, 1941-
Proving programs correct.

Bibliography: p.
Includes index.
1. Computer programs–Testing. 2. Debugging in
computer science. I. Title.
QA76.6.A47 001.6'425 78-9321
ISBN 0-471-03395-2

Printed in the United States of America
10 9 8 7 6 5 4 3 2

PREFACE

The purpose of this book is to explain and illustrate some basic techniques for proving computer programs correct. A large research effort has been devoted to this topic in recent years. Much of this research is aimed at formalizing and ultimately mechanizing such proofs. Our emphasis, however, is on rather informal correctness proofs of the type programmers can employ in trying to systematically convince themselves of their program's correctness. Of course, we are well aware that informal correctness proofs can easily contain errors and are no panacea for preventing or discovering all programming errors. Nevertheless, we do believe that such informal correctness proofs provide programmers with a more systematic means of desk checking their programs. We also think that the basic techniques used in correctness proofs give additional insights into the most basic programming constructs, looping, and recursion. For these reasons we believe that all programmers should be taught the basic techniques for proving programs correct.

The only prerequisites for understanding this book are programming experience in a high-level programming language and some slight exposure to mathematical proofs. Almost no specific mathematical knowledge is required. Chapter 1 provides a thorough introduction to mathematical induction, which is the main mathematical proof technique that underlies correctness proofs.

Chapter 2 examines the method of inductive assertions, which is the most commonly used technique in correctness proofs for iterative programs. It contains the most basic material. Chapter 3 illustrates further the method of inductive assertions for FORTRAN and

PL/I programs. It also briefly introduces the idea of verification rules and their equivalence with the method of inductive assertions. In Chapter 4 we deal with the method of structural induction, which is the most commonly used technique for proving the correctness of recursive programs. This technique is also shown to be useful in proving the correctness of iterative programs that are basically carrying out recursive processes. Chapter 5 is a very brief introduction to some of the current research related to proving program correctness. We have also supplied a fairly extensive bibliography as an aid to the reader who is interested in pursuing this topic.

This book can be used as a supplementary text for an undergraduate or first-year graduate course on the theory of computation. It can also be used as a supplementary text in a second course on programming emphasizing such topics as programming style and program correctness. Since it is sufficiently self-explanatory, it can also be used for self-study by any one with a slight familiarity with mathematical proofs and a background in programming.

Robert B. Anderson

CONTENTS

CHAPTER ONE

MATHEMATICAL INDUCTION

1.1 INTRODUCTION

Mathematical induction is a standard method of proof in mathematics. Although not always explicitly stated, it is the underlying technique of all correctness proofs for computer programs. This chapter is intended to thoroughly familiarize the reader with this fundamental method of proof.

Mathematical induction is usually stated as a method of proving statements about the positive integers. In the next section we state and illustrate the most elementary version of this method. In Section 1.3 we give a slightly stronger version of it, and in Section 1.4 we give a generalization of the method that is applicable to proving statements about any well-ordered set rather than just the positive integers. Only the material in Section 1.2 is necessary for most of the book. Therefore, you may prefer to skip Sections 1.3 and 1.4 and only return to them later if needed. Section 1.4 is more abstract than Sections 1.2 and 1.3 and should be skipped by the reader who lacks "mathematical maturity."

1.2 SIMPLE INDUCTION

THE SIMPLE INDUCTION PRINCIPLE

Suppose S (n) is some statement about the integer n and we wish to prove that S (n) is true for all positive integers n. The method of simple induction states that in order to prove this we only need to:

(i) Prove that S (1) is true.

(ii) Prove (for all positive integers n) that if S (n) is true then S (n + 1) is also true.

The fact that these two statements together do show that S (n) is true for all positive integers is intuitively obvious (although, in an axiomatic treatment of the integers, some form of this principle would have to be assumed as an axiom). From (i), we know that S(1) is true. From (ii) we know that if S(1) is true then S(2) is also true. But S(1) is true and hence S(2) must also be true. From (ii) we also know that if S(2) is true then S(3) is also true. Thus, since we know that S(2) is true, it follows that S(3) is also true, and so on. Hence, intuitively we see that (i) and (ii) together show that S(1), S(2), S(3),..., S (n),... are all true.

We now give several examples of the use of a simple induction.

EXAMPLE 1.2.1

We wish to prove for all positive integers n that the sum of the first n positive integers is equal to $n \cdot (n + 1)/2$. In other words, for all positive integers n, $1+2+\ldots+n = n\cdot(n + 1)/2$. To prove this by simple induction, we only need to prove:

(i) The sum of the first 1 positive integers is equal to $1\cdot(1+1)/2$, that is, $1 = 1\cdot(1+1)/2$. This is obviously true.

(ii) If the sum of the first n positive integers equals $n\cdot(n+1)/2$, then the sum of the first n+1 positive integers equals $(n+1)\cdot[(n+1)+1]/2$. Thus we may assume that $1+2+\cdots+n=n\cdot(n+1)/2$ is true. This is called the induction hypothesis, and we must try to prove that it follows from this that $1+2+\cdots+n+(n+1) = (n+1)\cdot[(n+1)+1]/2$ is also true.
To prove this note that

$$\begin{aligned}1+2+\cdots+n+(n+1) &= (1+2+\cdots n)+(n+1)\\ &= [n\cdot(n+1)/2]+(n+1) \quad \text{by the induction hypothesis}\\ &= [(n^2+n)/2]+(n+1)\\ &= [(n^2+n)/2]+[(2n+2)/2]\end{aligned}$$

$$= (n^2+3n+2)/2$$

$$= (n+1)\cdot(n+2)/2$$

$$= (n+1)\cdot[(n+1)+1]/2$$

This concludes the proof of part (ii). Since (i) and (ii) have both been proven, simple induction justifies the claim that for all positive integers n, $1+2+\cdots+n = n\cdot(n+1)/2$.

THE MODIFIED SIMPLE INDUCTION PRINCIPLE

Sometimes we wish to prove that a statement S(n) is true for all integers $n \geqq n_0$. Simple induction can be trivially modified to show this as follows. In order to prove that S(n) is true for all integers $n \geqq n_0$ we only need to:

(i) Prove that $S(n_0)$ is true.

(ii) Prove (for all integers $n \geqq n_0$) that if S(n) is true, then S(n+1) is also true.

In particular, if we wish to prove that some statement S(n) is true for all nonnegative integers (i.e., $n \geqq 0$), we only need to:

(i) Prove that S(0) is true.

(ii) Prove that (for all nonnegative integers n) if S(n) is true then S(n+1) is also true.

EXAMPLE 1.2.2

For all nonnegative integers n, we wish to prove that $2^0+2^1+2^2+\cdot+2^n = 2^{n+1} - 1$. In order to prove this by simple induction, do the following.

(i) Prove that $2^0 = 2^{0+1}-1$. But this is obvious, since $2^0 = 1 = 2^{0+1}-1$
$$= 2^1-1$$
$$= 2-1$$
$$= 1$$

(ii) Prove that (for all nonnegative integers n) if

$$2^0+2^1+2^2+\cdots+2^n = 2^{n+1}-1 \text{ is true, then}$$

$2^0+2^1+2^2+\cdots+2^n+2^{n+1} = 2^{(n+1)+1}-1$ is also true.

The statement $2^0+2^1+2^2+\cdots+2^n = 2^{n+1}-1$ is called the induction hypothesis. To prove (ii) note that

$$\begin{aligned} 2^0+2^1+2^2+\cdots+2^n+2^{n+1} &= (2^0+2^1+2^2+\cdots+2^n)+2^{n+1} \\ &= (2^{n+1}-1)+2^{n+1} \quad \text{by the induction hypothesis} \\ &= (2^{n+1}+2^{n+1})-1 \\ &= (2\cdot 2^{n+1})-1 \\ &= 2^{n+2}-1 \\ &= 2^{(n+1)+1}-1 \end{aligned}$$

We sometimes wish to prove that a statement S(n) is true for all integers n such that $n_0 \leqq n \leqq m_0$. Since there are only a finite number of integers between n_0 and m_0, we may be able to prove S(n) is true for all of these by merely checking all of the different cases. However, it is often easier and sometimes necessary (e.g., when we don't know specific values for n_0 or m_0) to prove S(n) by induction. In this situation there are two versions of simple induction that one can try to use to show that S(n) is true for all $n_0 \leqq n \leqq m_0$:

SIMPLE UPWARD INDUCTION

(i) Prove that $S(n_0)$ is true.

(ii) Prove (for all integers $n_0 \leqq n \leqq m_0-1$) that if S(n) is true then S (n+1) is also true.

SIMPLE DOWNWARD INDUCTION

(i) Prove that $S(m_0)$ is true.

(ii) Prove (for all integers $n_0+1 \leqq n \leqq m_0$ that if S(n) is true then S(n-1) is also true.

The student should be able to see that intuitively each of these is sufficient to prove that S(n) is true for all $n_0 \leqq n \leqq m_0$.

PROVING STATEMENTS ABOUT COMPUTER PROGRAMS

Sometimes it is ambiquous whether we are trying to prove that S(n) is true for all n so that $n_0 \leqq n \leqq m_0$ or $n_0 \leqq n$. In such situations we can frequently prove the result without knowing which of the two cases is involved. For example, in proving program correctness, we sometimes want to prove that a statement S is true each time execution reaches a particular point in the program. We might try to prove this by induction on n the number of times that execution has reached the point. But we may not know exactly how many times execution will reach this point - this may depend on what data is used when the program is executed. Execution may reach the point some finite number of times m_0, or it may reach the point an infinite number of times if the program fails to terminate. Thus we may be trying to prove that S(n) is true for all n so that $1 \leqq n \leqq m_0$ or $1 \leqq n$. Nevertheless, we may be able to prove the result without knowing which of the two possibilities is in fact the case. If we can prove the following, then we are justified in claiming that S(n) is true each time execution reaches the point:

(i) S(1) is true (i.e., S is true the first time execution reaches the point).

(ii) If S(n) is true (i.e., S is true the n^{th} time execution reaches the point) and execution returns to the point for an $n+1^{th}$ time, then S(n+1) is also true (i.e., S is true the $n+1^{th}$ time execution reaches the point)

If execution only reaches the point m_0 times, the only values of n for which the hypothesis of (ii) can possibly be true -- those values of n for which execution will return to the point for an $n+1^{th}$ time -- are all those values of n such that $1 \leqq n \leqq m_0-1$. On the other hand, if execution reaches the point an infinite number of times, the values of n for which the hypothesis of (ii) could be true are all those values of n such that $1 \leqq n$. Thus if we can prove (i) and (ii) we will have proved by simple upward induction or simple induction that S(n) is true for all the relevant values of n, regardless of which of the two possibilities is the case.

EXERCISES

1. Prove that for all positive integers n,

$$\frac{1}{(1)\cdot(2)} + \frac{1}{(2)\cdot(3)} + \cdots + \frac{1}{(n)\cdot(n+1)} = \frac{n}{n+1}$$

2. Prove that for all nonnegative integers n

$$3^0+3^1+3^2+\cdots+3^n = \frac{3^{n+1}-1}{2}$$

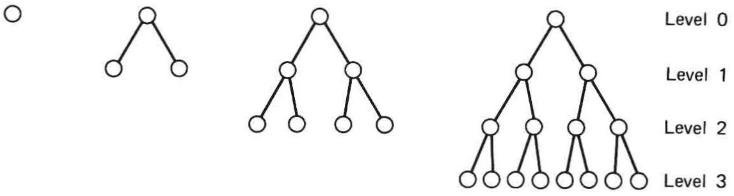
3. Prove that for all positive integers n,

$$1^3+2^3+\cdots+n^3 = \frac{n^2\cdot(n+1)^2}{4}$$

Note that this together with Example 1.2.1 proves the remarkable fact that

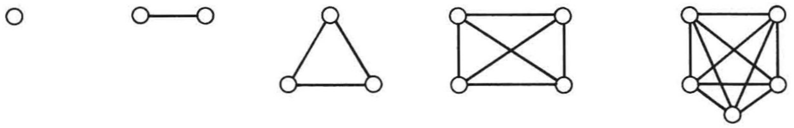
$$1^3+2^3+\cdots+n^3 = (1+2+\cdots+n)^2.$$

4.



The above graphs are examples of complete binary trees of levels 0, 1, 2, and 3. A complete binary tree of level n is a graph like the above in which all nodes except those on level n have two branches coming out of them. The nodes at level n that do not have any branches coming out of them are called the tip or leaf nodes of the tree. Prove by induction on the level n that the number of tip nodes in a complete binary tree of level n is 2^n. Find a formula for the total number of nodes (both tip and nontip nodes) in such a tree and prove the formula by induction.

5.



The above graphs are examples of complete graphs containing 1, 2, 3, 4, and 5 nodes. A complete graph on n nodes is a graph like the above, which contains n nodes and has one link or branch connecting each pair of nodes in the graph. Figure out a formula for the number of branches or links that occur in a complete graph on n nodes and prove the formula by induction on n.

6. Find the error in the following proposed proof. We wish to prove that

$$\frac{1}{1\cdot 2} + \frac{1}{2\cdot 3} + \cdots + \frac{1}{(n-1)\cdot(n)} = \frac{3n-2}{2n}$$

for all positive integers n. The proof is by induction on n.

(i) For n = 1, the formula is true for

$$\frac{1}{1\cdot 2} = \frac{3\cdot 1-2}{2\cdot 1} = \frac{3-2}{2} = \frac{1}{2}$$

(ii) Suppose the formula is true for n, that is,

$$\frac{1}{1\cdot 2} + \frac{1}{2\cdot 3} + \cdots + \frac{1}{(n-1)\cdot(n)} = \frac{3n-2}{2n}$$

then note that

$$\frac{1}{1\cdot 2} + \frac{1}{2\cdot 3} + \cdots + \frac{1}{(n-1)\cdot(n)} + \frac{1}{n(n+1)}$$

$$= \frac{1}{1\cdot 2} + \frac{1}{2\cdot 3} + \cdots + \frac{1}{(n-1)\cdot(n)} + \frac{1}{n(n+1)}$$

$$= \frac{3n-2}{2n} + \frac{1}{n(n+1)} \quad \text{by the induction hypothesis}$$

$$= \frac{(3n-2)(n+1)}{(2n)(n+1)} + \frac{2}{(2n)(n+1)}$$

$$= \frac{3n^2 + n - 2 + 2}{2n(n+1)}$$

$$= \frac{3n^2 + n}{2n(n+1)}$$

$$= \frac{n(3n+1)}{2n(n+1)}$$

$$= \frac{3n+1}{2(n+1)}$$

$$= \frac{3(n+1)-2}{2(n+1)}$$

Thus it is true for n+1 also.
Although the proof appears to be valid it must be incorrect, since when n = 4, we get

$$\frac{1}{1\cdot2} + \frac{1}{2\cdot3} + \frac{1}{3\cdot4} = \frac{1}{2} + \frac{1}{6} + \frac{1}{12} = \frac{9}{12} = \frac{3}{4}$$

$$\text{But } \frac{3n-2}{2n} = \frac{3\cdot4-2}{2\cdot4} = \frac{10}{8} \neq \frac{3}{4}$$

7. Find the error in the following proposed proof. We wish to prove that any collection of marbles contains only marbles of the same color. The proof will be by induction on the number, n, of marbles in the collection.

(i) For n = 1 it is obvious, since any collection of marbles that contains only one marble obviously contains only marbles of the same color.

(ii) Suppose the statement is true for any collection of n marbles. Let us show that it is then also true for any collection of n+1 marbles.

Suppose we picture a collection of n+1 marbles as

```
0 0 . . . 0   0
1 2       n  n+1
```

If we remove the n+1st marble from this collection, we are left with a collection of n marbles:

```
0 0 . . .  0
1 2        n
```

By the induction hypothesis all the marbles in this collection must be of the same color. Now suppose that we instead remove the first marble from the collection. Then we are left with the collection:

```
0 0 . . . 0   0
2 3       n  n+1
```

But this collection also contains n marbles and hence, by the induction hypothesis, all the marbles in this collection must also be of the same color. This implies that all n+1 marbles are of the same color, since we know that marbles

```
0 0 . . .  0
1 2        n
```

are all of the same color and the n+1st marble is also of the same color as marble n (in fact, it is not only the same color as marble n: it is the same color as marbles 2, 3, ..., n.) Thus all n+1 marbles are of the same color.

1.3 A STRONGER VERSION OF MATHEMATICAL INDUCTION

Sometimes a slightly stronger version of the induction method is needed to prove some statement about the integers. This slightly stronger version is the following.

THE STRONG INDUCTION PRINCIPLE

Suppose S(n) is some statement about the integer n and we wish to prove that S(n) is true for all positive integers n. In order to prove this we only need to:

(i) Prove that S(1) is true.
(ii) Prove (for all positive integers n) that if S(1), S(2), ..., S(n) are all true, then S(n+1) is also true.

Note that this stronger version of induction is identical to simple induction, except that in proving (ii) we get to assume as the induction hypothesis that all of the statements S(1), S(2), ..., S(n) are true rather than simply that S(n) is true. From this stronger induction hypothesis, we still need only to show that S(n+1) is true.

As with simple induction, it is intuitively clear that (i) and (ii) together imply that S(n) is true for all positive integers n. By (i) we know that S(1) is true. From (ii) we know that if S(1) is true, then S(2) is also true and hence, since S(1) is known to be true, then S(2) must also be true. But then, since S(1) is known to be true and S(2) is also known to be true, (ii) would imply that S(3) is also true. And, since S(1), S(2), and S(3) are all known to be true, (ii) would imply that S(4) is also true, etc.

We now give several examples where this stronger version of induction is useful.

EXAMPLE 1.3.1

A positive integer is called a prime number if the only positive integers that divide it without remainder are 1 and itself. We wish to prove that every positive integer n can be expressed as the product of (one or more) prime numbers. The proof is by strong induction on n.

(i) If n = 1, then it is itself a prime number and hence can be expressed as the product of (one) prime number(s).
(ii) Suppose that each of the numbers 1, 2, ...,n can be expressed as the product of prime numbers. We need to show that n+1 can also