94

How to Write a Proof

Leslie Lamport

February 14, 1993

Systems Research Center

DEC's business and technology objectives require a strong research program. The Systems Research Center (SRC) and three other research laboratories are committed to filling that need.

SRC began recruiting its first research scientists in 1984—their charter, to advance the state of knowledge in all aspects of computer systems research. Our current work includes exploring high-performance personal computing, distributed computing, programming environments, system modelling techniques, specification technology, and tightly-coupled multiprocessors.

Our approach to both hardware and software research is to create and use real systems so that we can investigate their properties fully. Complex systems cannot be evaluated solely in the abstract. Based on this belief, our strategy is to demonstrate the technical and practical feasibility of our ideas by building prototypes and using them as daily tools. The experience we gain is useful in the short term in enabling us to refine our designs, and invaluable in the long term in helping us to advance the state of knowledge about those systems. Most of the major advances in information systems have come through this strategy, including time-sharing, the ArpaNet, and distributed personal computing.

SRC also performs work of a more mathematical flavor which complements our systems research. Some of this work is in established fields of theoretical computer science, such as the analysis of algorithms, computational geometry, and logics of programming. The rest of this work explores new ground motivated by problems that arise in our systems research.

DEC has a strong commitment to communicating the results and experience gained through pursuing these activities. The Company values the improved understanding that comes with exposing and testing our ideas within the research community. SRC will therefore report results in conferences, in professional journals, and in our research report series. We will seek users for our prototype systems among those with whom we have common research interests, and we will encourage collaboration with university researchers.

Robert W. Taylor, Director

How to Write a Proof

Leslie Lamport

February 14, 1993

Author's Abstract

A method of writing proofs is proposed that makes it much harder to prove things that are not true. The method, based on hierarchical structuring, is simple and practical.

Contents

1 Mathematical Proofs

Mathematical notation has improved over the past few centuries. In the seventeenth century, a mathematician might have written

> There do not exist four positive integers, the last being greater than two, such that the sum of the first two, each raised to the power of the fourth, equals the third raised to that same power. (1)

How much easier it is to read the modern version

> There do not exist positive integers x, y, z, and n, with $n > 2$, such that $x^n + y^n = z^n$. (2)

Yet, the structure of mathematical proofs has not changed in 300 years. The proofs in Newton's *Principia* differ in style from those of a modern textbook only by being written in Latin. Proofs are still written like essays, in a stilted form of ordinary prose.

Formulas written in prose, like (1), are hard to understand and hard to get right. Proofs written in prose are also hard to understand and hard to get right. Anecdotal evidence suggests that as many as a third of all papers published in mathematical journals contain mistakes—not just minor errors, but incorrect theorems and proofs.

Statement (2) is easier to read than statement (1) for two reasons: variables are given names, and formulas are written in a more structured fashion. The benefits of using names is obvious. The benefit of structure is less obvious; we are so used to formulas like $x^n + y^n = z^n$ that we tend to take their structure for granted, and to think they are easy to read just because they are short. Although the brevity of the formula helps, it is primarily its structure that makes it easier to understand than a prose version. The expression

$$\begin{array}{lll} x \text{ raised to the power } n & & \\ \qquad\quad \textbf{plus} & & \\ y \text{ raised to the power } n & \quad\textbf{equals}\quad & z \text{ raised to the power } n \end{array}$$

is quite long, but it is easy to read because of its structure.

The same principles that make formulas easier to understand can make proofs easier to understand: proof steps should be referred to by name, and the structure of the proof should be manifest.

The proof style I advocate is a refinement of one, called *natural deduction*, that has been used by some logicians for almost a century. Natural deduction has been viewed primarily as a method of writing proofs in a formal logic. What I will describe is a practical method for writing the less formal proofs of ordinary mathematics. It is based on hierarchical structuring—a successful tool for managing complexity.

Avoiding mistakes when manipulating formulas requires careful, detailed calculations. Avoiding mistakes when proving theorems requires careful, detailed proofs. When first shown a detailed, structured proof, most mathematicians react: "I don't want to read all those details; I want to read only the general outline and perhaps some of the more interesting parts." My response is that this is precisely why they want to read a hierarchically structured proof. The high-level structure provides the general outline; readers can look at as much or as little of the lower-level detail as they want. However, until one gets used to them, structured proofs do look intimidating.

The ideal tool for reading a structured proof would be a computer-based hypertext system. It would allow the reader to concentrate on a particular level in the structure, suppressing lower-level details. In a printed version, one can ignore lower-level details only by skipping over that part of the text. While this is not ideal, the structure is displayed by the format, making such skipping fairly easy—certainly much easier than in a prose-style proof, where the format provides little clue to the logical structure.

2 An Example

I have discovered a remarkable proof of (2), but it is too long to use here as an example. Instead, I take as an example the classic proof that $\sqrt{2}$ is irrational. Letting $\mathbf{Q}$ denote the set of rationals, the precise statement of the result to be proved is

> **Theorem** There does not exist r in $\mathbf{Q}$ such that $r^2 = 2$.

To illustrate hierarchical structure, the proof is carried out to a much lower level of detail than necessary for a typical reader.

2.1 The High-Level Proof

The high-level structure of the proof—what one would see first with a hypertext system—appears in Figure 1. The proof assumes a lemma from which one can deduce that, for any integer n, if 2 divides n^2 then 2 divides n. The set of integers is denoted by $\mathbf{Z}$.

Theorem There does not exist r in $\mathbf{Q}$ such that $r^2 = 2$.

Proof sketch: We assume $r^2 = 2$ for $r \in \mathbf{Q}$ and obtain a contradiction. Writing $r = m/n$, where m and n have no common divisors (step 1), we deduce from $(m/n)^2 = 2$ and the lemma that both m and n must be divisible by 2 (steps 2 and 3).

Assume: 1. $r \in \mathbf{Q}$
2. $r^2 = 2$

Prove: False

1. Choose m, n in $\mathbf{Z}$ such that
   1. $\gcd(m, n) = 1$
   2. $r = (m/n)$
2. 2 divides m.
3. 2 divides n.
4. Q.E.D.

Figure 1: The highest level of a structured proof of the irrationality of $\sqrt{2}$.

After the statement of the theorem comes a Proof Sketch, which is an informal explanation of the following proof. The proof sketch serves as a "road map" to the proof, helping the reader understand intuitively why the proof works. This proof is so simple that the proof sketch is almost superfluous—the only information it provides that is not obvious from the high-level proof itself is that the lemma is used to prove steps 2 and 3.

Next comes the Assume and Prove clauses. They assert that to prove the theorem, it suffices to assume the two hypotheses $r \in Q$ and $r^2 = 2$, and to prove *false*.

Finally comes the proof. This is a sequence of statements that ends with "Q.E.D.", which denotes the assertion to be proved—in this case, *false*. Think of this proof as the left half (the statements) of a high-school geometry style proof, the right half (the reasons) being omitted.[1]

2.2 Lower Levels of the Proof

Let us now examine the proof of step 1, which appears in Figure 2. It is clear enough what must be proved, so no Assume/Prove is needed. The proof consists of five steps, numbered 1.1 through 1.5. There is also a Let

[1] In their introductory plane geometry course, American students are taught to write proofs in a two-column format, the left column containing a sequence of statements and the right column containing their justifications.

```
1. Choose $m$, $n$ in $\mathbf{Z}$ such that
   1. $\gcd(m, n) = 1$
   2. $r = (m/n)$
  1.1. Choose $p$, $q$ in $\mathbf{Z}$ such that $q \neq 0$ and $r = p/q$.
  Let: $m \triangleq p/\gcd(p, q)$
       $n \triangleq q/\gcd(p, q)$
  1.2. $m, n \in \mathbf{Z}$
  1.3. $r = m/n$
  1.4. $\gcd(m, n) = 1$
  1.5. Q.E.D.
```

Figure 2: The proof of Step 1.

statement, which defines the required m and n. (I prefer $\triangleq$ to the more common symbol $\equiv$ for "equals by definition", since $\equiv$ can also mean logical equivalence.)

Each of these five steps in turn has its proof. The proof of 1.1 is just

> Proof: By assumption :1.

Assumption :1 is the first assumption ($r \in \mathbf{Q}$) in the proof of the theorem. (The numbering scheme for assumptions is explained below.) A hierarchical proof must stop somewhere. The general question of where to stop is addressed in Section 4.2. In this proof, we assume the reader understands that the definition of $\mathbf{Q}$ implies that r can be written as the requisite quotient of integers. The proof of 1.2 is the equally simple.

> Proof: 1.1 and definition of m and n.

Step 1.3 is proved by a string of equalities, each with a brief justification.

$$\begin{aligned} \text{Proof: } m/n &= \frac{p/\gcd(p,q)}{q/\gcd(p,q)} && \text{[Definition of } m \text{ and } n\text{]} \\ &= p/q && \text{[Simple algebra]} \\ &= r && \text{[By 1.1]} \end{aligned}$$

This type of proof, consisting of a string of equalities, is simple and direct; it works as well for proving any transitive relation, such as <, logical equivalence, and implication. It should be used whenever possible.

Step 1.4 has the multistep proof shown in Figure 3, consisting of steps 1.4.1 through 1.4.3. The "1.4:1" in the proof of step 1.4.1 denotes assumption 1 (s divides m) in the proof of step 1.4. The theorem itself is considered

1.4. $\gcd(m, n) = 1$

Proof: By the definition of the gcd, it suffices to:

Assume: 1. s divides m

2. s divides n

Prove: $s = 1$

1.4.1. $s \cdot \gcd(p, q)$ divides p.

Proof: 1.4:1 and the definition of m.

1.4.2. $s \cdot \gcd(p, q)$ divides q.

Proof: 1.4:2 and definition of n.

1.4.3. Q.E.D.

Proof: 1.4.1, 1.4.2, and the definition of gcd.

Figure 3: The proof of step 1.4.

.

to be a step having the null string as its number, which explains why ":1" denotes assumption 1 of the theorem.

3 Further Details

3.1 A More Compact Numbering Scheme

The numbering scheme used in the example is fine for short proofs, with few levels of nesting. However, long proofs can have many levels—I often write proofs more than six levels deep. The number 3.1.1.1.1.2 takes a lot of space, and having to distinguish it from 3.1.1.1.2 can soon lead to eye strain.

We eliminate long step numbers by abbreviating 3.1.1.1.2, a five-part step number ending in 2, as $\langle 5\rangle 2$. Figure 4 shows a fragment of a proof written with the two numbering styles. To understand why abbreviated numbers suffice, consider where step 3.1.1.1.2 can be used in this proof. The step can be used only after it is proved, but it cannot be used anywhere after its proof. Step 3.1.1.1.2 cannot be used in the proof of step 3.1.1.2 because it was proved under the assumption of step 3.1.1.1, which is different from step 3.1.1.2's assumption. The step can be used only where the assumptions under which it was proved hold, which means that it can be used only within the proof of its parent, step 3.1.1.1. Step 3.1.1.1.2 is the only one in the proof of its parent with a five-part number ending in 2. Although there can be many proof steps with the same abbreviated number $\langle 5\rangle 2$, no two of them

3.1.1.1. Assume: $x \in S$
Prove: ...
3.1.1.1.1. ...
3.1.1.1.2. ...
3.1.1.1.3. Q.E.D.
By 3.1.1.1.1 and assumption 3.1.1.1.
3.1.1.2. Assume: $x \in T$
Prove: ...
...

$\langle 4\rangle 1$. Assume: $x \in S$
Prove: ...
$\langle 5\rangle 1$. ...
$\langle 5\rangle 2$. ...
$\langle 5\rangle 3$. Q.E.D.
By $\langle 5\rangle 1$ and assumption $\langle 4\rangle$.
$\langle 4\rangle 2$. Assume: $x \in T$
Prove: ...
...

Figure 4: Part of a proof, with long and abbreviated step numbers.

have the same parent, so at most one of them may be used at any point in the proof. A reference to step $\langle 5\rangle 2$ always refers to the most recent step with that number. Part 3 of the statement of step $\langle 5\rangle 2$ is numbered $\langle 5\rangle 2.3$.

References to assumptions can be abbreviated even more. An assumption can be used only in the proof of a step, or the proof of one of its descendants. We let $\langle 5\rangle$ denote the assumption of the level-five step that is an ancestor of (or is) the current step, and $\langle 5\rangle{:}3$ denote the third numbered part of that assumption. Since the statement of the theorem has a zero-part number, its assumption is number $\langle 0\rangle$.

Figure 5 contains the complete proof of our example, written with the abbreviated numbering scheme.

3.2 Proof by Cases

Proof by cases can be expressed with a Case step, where

Case: Statement of assumption.

is an abbreviation for

Assume: Statement of assumption.

Prove: Q.E.D.

The proof of the final "Q.E.D." step explains why the cases considered are exhaustive; it is usually simple. Figure 6 illustrates the use of the Case construct to structure a proof by induction. Note how step $\langle 1\rangle 1$ is used in the proofs of both cases, showing why Case steps provide more flexibility than would a strictly hierarchical proof-by-cases construct.

Theorem There does not exist r in $\mathbf{Q}$ such that $r^2 = 2$.

PROOF SKETCH: We assume $r^2 = 2$ for $r \in \mathbf{Q}$ and obtain a contradiction. Writing $r = m/n$, where m and n have no common divisors (step $\langle 1 \rangle 1$), we deduce from $(m/n)^2 = 2$ and the lemma that both m and n must be divisible by 2 ($\langle 1 \rangle 2$ and $\langle 1 \rangle 3$).

ASSUME: 1. $r \in \mathbf{Q}$
2. $r^2 = 2$

PROVE: False

$\langle 1 \rangle 1$. Choose m, n in $\mathbf{Z}$ such that
1. $\gcd(m, n) = 1$
2. $r = (m/n)$

$\langle 2 \rangle 1$. Choose p, q in $\mathbf{Z}$ such that $q \neq 0$ and $r = p/q$.
PROOF: By assumption $\langle 0 \rangle$:1.

LET: $m \triangleq p/\gcd(p, q)$
$n \triangleq q/\gcd(p, q)$

$\langle 2 \rangle 2$. $m, n \in \mathbf{Z}$
PROOF: $\langle 2 \rangle 1$ and definition of m and n.

$\langle 2 \rangle 3$. $r = m/n$
PROOF:
$$\begin{aligned} m/n &= \frac{p/\gcd(p,q)}{q/\gcd(p,q)} && \text{[Definition of } m \text{ and } n\text{]} \\ &= p/q && \text{[Simple algebra]} \\ &= r && \text{[By } \langle 2 \rangle 1\text{]} \end{aligned}$$

$\langle 2 \rangle 4$. $\gcd(m, n) = 1$
PROOF: By the definition of the gcd, it suffices to:
ASSUME: 1. s divides m
2. s divides n
PROVE: $s = 1$

$\langle 3 \rangle 1$. $s \cdot \gcd(p, q)$ divides p.
PROOF: $\langle 2 \rangle$:1 and the definition of m.

$\langle 3 \rangle 2$. $s \cdot \gcd(p, q)$ divides q.
PROOF: $\langle 2 \rangle$:2 and definition of n.

$\langle 3 \rangle 3$. Q.E.D.
PROOF: $\langle 3 \rangle 1$, $\langle 3 \rangle 2$, and the definition of gcd.

$\langle 2 \rangle 5$. Q.E.D.

$\langle 1 \rangle 2$. 2 divides m.

$\langle 2 \rangle 1$. $m^2 = 2n^2$
PROOF: $\langle 1 \rangle 1.1$ implies $(m/n)^2 = 2$.

$\langle 2 \rangle 2$. Q.E.D.
PROOF: By $\langle 2 \rangle 1$ and the lemma.

Figure 5: A proof of the irrationality of $\sqrt{2}$.

$\langle 1 \rangle 3.$ 2 divides n.

$\langle 2 \rangle 1.$ Choose p in $\mathbf{Z}$ such that $m = 2p$.
Proof: By $\langle 1 \rangle 2$.

$\langle 2 \rangle 2.$ $n^2 = 2p^2$
Proof:
$$\begin{aligned} 2 &= (m/n)^2 && [\langle 1 \rangle 1.2 \text{ and } \langle 0 \rangle{:}2] \\ &= (2p/n)^2 && [\langle 2 \rangle 1] \\ &= 4p^2/n^2 && [\text{Algebra}] \end{aligned}$$
from which the result follows easily by algebra.

$\langle 2 \rangle 3.$ Q.E.D.
Proof: By $\langle 2 \rangle 2$ and the lemma.

$\langle 1 \rangle 4.$ Q.E.D.
Proof: $\langle 1 \rangle 1.1$, $\langle 1 \rangle 2$, $\langle 1 \rangle 3$, and definition of gcd.

Figure 5 (continued)

Theorem All natural numbers are interesting.
Assume: n a natural number.
Prove: n is interesting.

$\langle 1 \rangle 1.$ A number is interesting if it is the smallest number not in an interesting set.
Proof: By definition of interesting.

$\langle 1 \rangle 2.$ Case: $n = 0$
Proof: By $\langle 1 \rangle 1$, since 0 is the smallest natural number not in $\emptyset$.

$\langle 1 \rangle 3.$ Case: 1. $n > 0$
2. $n - 1$ is interesting
Proof: By $\langle 1 \rangle 1$, since case assumption $\langle 1 \rangle$ implies that $\{k : k > n - 1\}$ is interesting.

$\langle 1 \rangle 4.$ Q.E.D.
Proof: Steps $\langle 1 \rangle 2$ and $\langle 1 \rangle 3$, assumption $\langle 0 \rangle$, and mathematical induction.

Figure 6: The Case construct.

4 How Good Are Structured Proofs?

4.1 My Experience

Some twenty years ago, I decided to write a proof of the Schroeder-Bernstein theorem for an introductory mathematics class. The simplest proof I could find was in Kelley's classic general topology text [4, page 28]. Since Kelley was writing for a more sophisticated audience, I had to add a great deal of explanation to his half-page proof. I had written five pages when I realized that Kelley's proof was wrong. Recently, I wanted to illustrate a lecture on my proof style with a convincing incorrect proof, so I turned to Kelley. I could find nothing wrong with his proof; it seemed obviously correct! Reading and rereading the proof convinced me that either my memory had failed, or else I was very stupid twenty years ago. Still, Kelley's proof was short and would serve as a nice example, so I started rewriting it as a structured proof. Within minutes, I rediscovered the error.

My interest in proofs stems from writing correctness proofs of algorithms. These proofs are seldom deep, but usually have considerable detail. Structured proofs provided a way of coping with this detail. The style was first applied to proofs of ordinary theorems in a paper I wrote with Martín Abadi [2]. He had already written conventional proofs—proofs that were good enough to convince us and, presumably, the referees. Rewriting the proofs in a structured style, we discovered that almost every one had serious mistakes, though the theorems were correct. Any hope that incorrect proofs might not lead to incorrect theorems was destroyed in our next collaboration [1]. Time and again, we would make a conjecture and write a proof sketch on the blackboard—a sketch that could easily have been turned into a convincing conventional proof—only to discover, by trying to write a structured proof, that the conjecture was false. Since then, I have never believed a result without a careful, structured proof. My skepticism has helped avoid numerous errors.

I have also found structured proofs very helpful when I need a variant of an existing theorem, perhaps with a slightly weaker hypothesis. In a properly written proof, where every use of an assumption or a proof step is explicit, simple text searching reveals exactly where every hypothesis is used.

4.2 Writing Structured Proofs

A structured proof format by itself will not eliminate errors. Proofs must be written carefully, with enough detail. Most errors come from not carrying out the proof to enough levels. The lowest-level, paragraph-style proofs should be short and completely transparent. One must be a skeptical reader of one's own proofs. My own rule of thumb is to expand the proof until the lowest level statements are obvious, and then continue for one more level. This takes discipline. But, unlike conventional proofs, in which adding more detail can make a proof more confusing, structured proofs accommodate as much detail as desired.

Structured proofs are longer than conventional ones. Although the formatting is partly responsible, structured proofs are longer mainly because they include more detail. They make it obvious when steps have been forgotten or important details omitted. They make it hard to be sloppy. The assertion "this case is similar to the previous one" is not acceptable; one is forced to find the appropriate general step that makes the proof of both cases easy. Writing a rigorous proof is harder than writing a sloppy one, and lazy writers will find excuses to avoid doing it. A common excuse is that structured proofs are too long. But, shorter proofs are not necessarily better ones; the shortest proof is always "left as an exercise for the reader."

When journals are distributed electronically, they can include proofs down to the lowest reasonable level; the reader can suppress uninteresting details when viewing the article on the screen or printing it locally. But, for paper journals, extra pages mean killing extra trees. It may be inappropriate for a journal to print a proof with so much detail. I recommend that authors provide two versions of their proofs: a very detailed one for themselves, the referees, and interested colleagues; and a less detailed one for paper publication. It is quite easy to convert a detailed proof into a less detailed one by compressing the lower levels into paragraph-style proofs. Although the reader must fill in the low-level details, such proofs are much better than unstructured ones, in which authors seem to choose randomly which details to supply and which to omit.

4.3 Reading Structured Proofs

So far, readers' reactions to structured proofs have been mixed. Skeptical readers—ones who check for errors—like these proofs much more than conventional ones. Readers who want to skim the proofs are less happy with the style. Part of the problem is that the length of the proofs and the unfamiliar format are intimidating. The best way to read a structured proof is level by level—first reading the high-level steps $\langle 1\rangle 1$, $\langle 1\rangle 2$, $\langle 1\rangle 3, \ldots$, then the proofs of those steps, and so on. However, having to skip over the lower-level steps makes reading the high-level ones inconvenient. With hypertext, this is not a problem. With printed text, a layered presentation may help [3, section B.7 (page 48)].

These structured proofs do not seem ideal for someone who wants to understand the important ideas of a proof without reading any of the details. Satisfying such readers may just require better proof sketches. Or, perhaps a better way of annotating a proof with comments is needed. Hypertext can provide graphical aids for finding one's way around a proof and highlighting important steps. Maybe such aids can be developed for the printed page.

4.4 The Future

Modern mathematical notation has evolved over hundreds of years. Its proof style is still stuck in the seventeenth century. Mathematicians tend to be conservative, and many are unwilling to consider that there might be a better way of writing proofs. But, I am told that mathematicians are embarrassed to learn that they published incorrect theorems, so they are motivated to avoid errors. I believe they will like structured proofs if they can be persuaded to try them.

Computer scientists are more willing to explore unconventional proof styles. Unfortunately, I have found that few of them care whether they have published incorrect results. They often seem glad that an error was not caught by the referees, since that would have meant one fewer publication. I fear that few computer scientists will be motivated to use a proof style that is likely to reveal their mistakes. Structured proofs are unlikely to be widely used in computer science until publishing incorrect results is considered embarrassing rather than normal.

The proof style described here has been developed over the past several years. I have written many hundreds of pages of structured proofs, mostly of algorithms. I consider the style to be a great improvement over conventional, unstructured proofs. But, this is not the last word on the subject. I look forward to seeing structured proof styles evolve as mathematicians and computer scientists find better ways to write a proof.

Acknowledgements

My information about mathematicians' errors and embarrassment comes mainly from George Bergman. The CASE construct and several other details of the proof format were developed in discussions with Urban Engberg and Peter Grønning.

References

[1] Martín Abadi and Leslie Lamport. Composing specifications. In J. W. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Stepwise Refinement of Distributed Systems*, volume 430 of *Lecture Notes in Computer Science*, pages 1–41. Springer-Verlag, May/June 1989.

[2] Martín Abadi and Leslie Lamport. The existence of refinement mappings. *Theoretical Computer Science*, 82(2):253–284, May 1991.

[3] Martín Abadi and Leslie Lamport. An old-fashioned recipe for real time. Research Report 91, Digital Equipment Corporation Systems Research Center, 1992.

[4] John L. Kelley. *General Topology*. The University Series in Higher Mathematics. D. Van Nostrand Company, Princeton, New Jersey, 1955.