A Note on "How to Write a Proof"

Piotr Rudnicki[*] Andrzej Trybulec[†]

January 13, 1996[‡]

Abstract

We believe that mechanical checking of real-life proofs can become practical and therefore we use MIZAR—a proof checking system for proofs written in a style of traditional mathematics. In the beginning of 1994 we came across a copy of L. Lamport's [8] paper in which "a method for writing proofs is proposed that makes it much harder to prove things that are not true." For MIZAR users the issue of *How to Write a Proof?* is an important one, as MIZAR is a proof checker and not an automated prover. We have tested MIZAR fitness for writing structured proofs in Lamport's style by rewriting his proof of the irrationality of $\sqrt{2}$ into MIZAR. It was not surprising to notice that formatting conventions help in presenting and reading proofs. However, such conventions do not, as they cannot, guarantee the correctness of the written proof, our little test being a case in point. We advocate development and employment of mechanical checkers for proofs.

1 Writing and reading proofs

L. Lamport [8] proposes a format for writing proofs of ordinary mathematics in a style of *natural deduction*. He claims that the prevailing practice of writing proofs "like essays, in a stilted form of ordinary prose" yields proofs which are "hard to understand and hard to get right". The basic principle of Lamport's method is: "proof steps should be referred to by name, and the structure of the proof should be manifest". His approach "is based on

[*]Supported in part by NSERC Grant No. OGP9207. Address: Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada T6J 2H1. e-mail: `piotr@cs.ualberta.ca`

[†]Supported in part by NSERC Grant No. OGP9207. Address: Institute of Mathematics, Warsaw University in Białystok, 15-267 Białystok, Poland. e-mail: `trybulec@math.uw.bialystok.pl`

[‡]Most of the work reported here was done in June 1994.

hierarchical structuring—a successful tool for managing complexity”. With a computer-based hypertext system one could read a structured proof at any recorded level of detail, suppressing lower level details. This is harder to do with a printed version but if the text format mirrors the logical structure of the proof it is relatively easy to skip over lower level details. Lamport illustrates his method on a proof that $\sqrt{2}$ is irrational (it is included in Appendix B as it appeared in [8]). We discuss this example below.

One should distinguish the process of discovering a proof from the process of presenting the proof. It is unlikely that any formatting conventions can help much with the former, while they can be of substantial assistance with the latter. Such conventions, as Lamport writes, can significantly help in reading a proof. But formatting conventions may clarify only certain aspects of a proof, namely the ones related to its logical structure. In informal proofs, nicely formatted or not, we tend to leave many things implicit, making tacit assumptions about a reader's knowledge and skills. Reading a proof becomes quite a challenge when these assumptions are not satisfied or misunderstood by the reader. Let us list some of the ‘challenges’ facing a reader of an informal proof:

- The logical structure of a proof must be recovered from a proof written in a paragraph style (this is the main point behind Lamport's paper).
- The overall proof structure can be presented in different ways. All of the following:

  let x be a natural number divisible by 7 ...
  consider x being a natural number divisible by 7 ...
  assume 7 divides x ...

  can be used at the beginning of a proof that “All numbers divisible by 7 are interesting”. However, the first and the second construct are also used to choose an object with some property if we know that it exists.

  The above is not a serious problem but it is a possible source of misunderstandings and one is tempted to ask: why do not we use a standard collection of proof structuring constructs?

- Some assumptions are not mentioned every time when they are used, they can be covered by a blanket statement pages away from the point of reading.
- Not all necessary premises are listed at an inference step.

- A reader is assumed to be skillful at following logical inferences of varying complexity. Many are frustrated by seeing neighboring inferences one of which is trivial and the other very complicated.
- References, also tacit ones, to theorems and definitions which may not be remembered by or known to the reader.
- Proofs contain errors. Even typographical errors hinder the process of reading a proof. However, one learns a lot from finding and especially from correcting an error.

The issues mentioned above make reading proofs a complex task. From this viewpoint, writing down a proof such that it is easy to follow becomes a time consuming and an error prone activity. Can these processes be assisted by a suitable software system? We believe that the answer is *yes*—although building such a system, even an experimental one, proved to be a task much bigger than we had originally envisaged. (The feasibility and the practicality of such systems is a subject of a debate, see [2, 4].)

For more than 20 years now, the second author has been leading the MIZAR project aimed at developing a software system for checking proofs written in the style of traditional mathematics. The system is meant to assure correctness of written proofs. Since the proofs are available in machine readable form, there is a possibility for their mechanical editing, pretty-printing, and access via hypertext based browsers. Such utilities can facilitate reading and learning proofs.

The key feature of the MIZAR system is that proofs are written by people and then checked for correctness by a machine. The proof authoring language, called the MIZAR language, has been designed as a formalized version of the traditional language of ordinary mathematics, tailored for automated processing. The logical structure of the language is based on a natural deduction system developed by Jaśkowski. The Tarski-Grothendieck set theory and axioms of the strong arithmetic of real numbers form the basis of doing mathematics in MIZAR. (One of the current efforts in the MIZAR project is to define arithmetic in set theory such that the arithmetic axioms will become theorems.)

The texts written in MIZAR take the form of articles and are organized into a data base called Mizar Mathematical Library (MML for short). The implemented processor of the language checks the articles for logical consistency and correctness of references to other articles. A short overview of the MIZAR project can be found in [11][1].

[1] See also: `http://www.cs.ualberta.ca/~piotr/Mizar`. For more documents on

When, in the beginning of 1994, we came across a printed copy of [8] it drew our immediate attention. The MIZAR processor is only a proof checker and requires quite detailed proofs in a rather demanding formalism. The question *how to format a proof?* is important for MIZAR authors as proofs in MIZAR need to be precise and sometimes tend to be long. However, it has been observed that MIZAR proofs, once completed and mechanically checked, are infrequently read as they are trusted to be correct, but nonetheless, they are sometimes read. The reading of MIZAR proofs is usually driven by curiosity and not necessity. Namely, in order to make references to theorems and definitions collected in MML one needs to know only the statements of the theorems and not their proofs.

The MIZAR input language does not impose any relationship between the format and the logical structure of a proof—the input to the processor is a plain text file. MIZAR authors use formatting conventions fitting their personal taste. It has been noticed that proof authors, working under the scrutiny of an automated proof checker, pay little attention to whether their proofs are easy to read by others or not. Frequently, the logical structure of a proof is not mirrored in the layout of the proof text. However, recovering this structure is assisted in part by using a text editor. Undoubtedly, a standard for formatting MIZAR proofs would benefit both authors of proofs and also the potential readers. Therefore, we have decided to try Lamport's proposal for formatting proofs by applying it to some MIZAR proofs.

2 An Example

We have experimented with proving the irrationality of $\sqrt{2}$ in two different ways. In our first approach, we decided to employ Lamport's formatting style in a proof of the irrationality of $\sqrt{2}$ in which we tried to use whatever relevant material was available in MML. The second, and more interesting, part of our experiment was to rewrite Lamport's proof from [8], quoted here in Appendix B, into MIZAR, without unnecessary departures from the idea or the structure of his proof.

MIZAR texts quoted in this note have been checked by MIZAR processor version 5.2.12.

MIZAR write to Roman Matuszewski (Mizar Users Group) email: `romat@plearn.edu.pl`.

2.1 Our proof

The complete MIZAR text of our proof[2] is contained Appendix A. We used the formalization of rational numbers as developed in [6]. First, we proved the following fact[3]:

```
Fact: not ex r being Rational st 0 ≤ r & r^2 = 2
```

with the help of which the proof of the main theorem was reduced to considering two simple cases.

We were lucky that MML already contained the following theorem:

```
theorem ::  NAT_LAT:95
for i being Nat st i is prime holds
   for m, n being Nat holds i | m · n implies (i | m or i | n)
```

which was proven by S. Białecki and included in an article by M. Chmur [3]. This theorem is a general version of the lemma assumed by Lamport for his proof. (The proof of this theorem is substantially more complicated than any of the proofs that we quote in the appendices: its proof in MIZAR is about 100 lines long.)

To summarize, we were quite pleased with this part of our experiment. Writing the proof took us about two hours with most of the time spent searching MML for needed facts.

2.2 Rewriting Lamport's proof

We have taken Lamport's proof from [8] (see Appendix B) at its face value: an exemplary proof written by an experienced proof writer that illustrates his method of writing proofs. This small experiment was meant to address the following questions:

- How long is Lamport's proof rewritten in MIZAR?

  The MIZAR proof checker is relatively weak and sometimes one is forced to record one's reasoning in very detailed steps in order to get it accepted. The proof checker is also uneven: sometimes it accepts inferences whose correctness is far from obvious.

---

[2] While formatting the proof we tried to follow Lamport's suggestions to the extent permitted by MIZAR syntax (some comments are given in Section 2.2).

[3] MIZAR texts are written in extended ASCII. In this paper, when we quote a fragment of a MIZAR text, some extended ASCII characters are transliterated into symbols available in TEX. We hope that the meaning of all symbols can be guessed from context.

- Is all the terminology needed to state and prove the theorem available in the MIZAR data base?

  Our experiment would definitely become a bigger work if MML did not include the definition of *gcd* or certain basic properties of *gcd*. MML is under continuous development; it is always possible that an author has to introduce new terminology or prove some auxiliary facts if they are not present in MML.

- How closely can we follow Lamport's proof format?

  We have used the hierarchical structuring when writing proofs for a long time but it never resulted in a stable set of formatting rules. When the proofs are automatically checked for correctness, an author is not pressed for a nice formatting style.

Lamport's proof has been translated into two short MIZAR texts: the main proof is in Appendix C and Appendix D contains an auxiliary text.

2.3 Labeling and formatting

The lexical restrictions on labels in MIZAR do not permit a direct use of the Lamport's numbering scheme. We have therefore adopted the following lexical conventions for labels:

- The label of an item k appearing at the nesting level n is written as `n'k` instead of $\langle n \rangle$k.

- The label of an assumption starts with an `A` and then a reference to the first assumption of the level-zero step is made by writing `A0'1` instead of $\langle 0 \rangle$:1.

- No part of a label can be omitted. In an assumption with multiple parts like:

  $\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

  all parts are labeled using fully expanded labels:

```
      let s be Integer such that
A2'1: s | m and
A2'2: s | n;
```

- Only formulae can be labeled in MIZAR. In consequence, the `consider` construction which corresponds to Lamport's "choose" cannot be labeled, and therefore Lamport's

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

  is translated to

```
     consider m, n being Integer such that
1'1_1: m gcd n = 1        and
1'1_2: r = m/n            and
1'1_3: n <> 0
```

  with the parts of formulae introduced by the `consider` construction labeled with fully expanded labels. The l-th part of step $\langle n \rangle$k is labeled `n'k_l`.

- Referring to entire constructs like "choose" may not help in reading proofs. Look at:

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

  By referring to $\langle 2 \rangle$1 we make a reference to

$$q \neq 0 \text{ and } r = p/q$$

  and the reader must analyze which of the conjuncts is relevant in a specific case. We were tempted to rewrite this step in MIZAR as:

```
     consider p, q being Integer such that
2'1_1: q <> 0  and
2'1_2: r = p/q  by A0'1, RAT_1:def 1;
```

  as it is better to read, but finally we decided not to change Lamport's format when we did not have to.

2.4 Environment

The MIZAR data base is built on top of a collection of MIZAR articles. When writing a new article we have to specify articles from which we want to import terminology, notations, theorems, definitions, etc. Certain terminology is built-in, such as the elementhood predicate $\in$ and the type `Real`. (Vocabulary symbols are imported from *vocabularies* which are separate from articles.)

MML is a result of contributions by many authors. The policy for accepting new articles has been quite liberal which has resulted in a "raw" repository of mathematical facts. Inevitably, in such a repository there are numerous repetitions and certain facts or notions are introduced in places that are hard to guess and thus hard to find. MML is undergoing a permanent amelioration.

Let us enumerate the terminology and notation used in the MIZAR text in Appendix C:

Built-in notions or imported by default:

- type `non empty set`,
- the elementhood relation $\in$,
- small natural constants: 0, 1, 2,
- type `Real`,
- equality, `=`, and inequality, `<>`

Notions imported explicitly:

- The set of rational numbers `RAT` from article `RAT_1` [6].
- The square function with standard notation from article `SQUARE_1` [12].
- Type `Integer` from article `INT_1` [13].
- The `gcd` function in infix notation for integer arguments from article `INT_2` [7].
- The division `/` and multiplication $\cdot$ for `Real`s from article `REAL_1` [5].
- The divisibility relation `|` for integer arguments from article `INT_1` [13].

In order not to obscure the main proof we have prepared an auxiliary MIZAR text, named `LAUX_0` (see Appendix D), which serves as a buffer between the task at hand and the current contents of MML. This file contains six simple facts about divisibility and the `gcd` function that one would expect to find in a data base of mathematical facts. A portion of the auxiliary file is related to resolving typing problems as explained below.

2.5 Typing problems

Lamport's proof is written using untyped objects and to say that r is rational he writes $r \in \mathbf{Q}$.

MIZAR supports typing of variables. Although the MIZAR language requires that every object be typed, one can imagine writing MIZAR texts using only the most general type, namely `set`. However, MIZAR authors usually introduce narrower types (sorts) organized into a tree-like hierarchy. The authors have complete freedom in introducing new types and operations on objects of these types. With this in mind, it was not surprising that in some cases the notions that we have imported have been typed differently than required by our needs.

The square function

In our proof we needed to talk about the divisibility of `i`^{2}, for integer `i`. However, the imported squaring functor (from article `SQUARE_1` [12]) results in a real number and the imported divisibility relation (from article `INT_1` [13]) requires integer arguments. This problem is easily resolved by employing the so called *functional cluster* construct of MIZAR:

```
definition let i be Integer;
 cluster i² -> integer;
 coherence proof
   i² = i·i by SQUARE_1:58;
  hence i² is integer;
 end;
end;
```

The `coherence` condition requires us to prove that indeed a square of an integer is an integer (it is really a natural number but we want to talk about integers here). The above definition was introduced in the auxiliary article, `LAUX_0`, whose abstract—statements of definitions and theorems without their proofs—is in Appendix D.

The divisibility relations

As we mentioned on page 5, MML contains a version of the lemma assumed in Lamport's proof. However, this fact (named `NAT_LAT:95`) was formulated in terms of the divisibility of natural numbers. MML, for our inconvenience, contained two definitions of divisibility, one for natural numbers (in article `NAT_1` [1]):

```
definition let k,l be Nat;
  pred k | l means
:: NAT_1: def 3
  ex t st l = k · t ;
end;
```

and a separate, unrelated, definition of divisibility for integers (in article INT_1 [13]):

```
 definition
  let i1,i2 be Integer;
  pred i1 | i2 means
:: INT_1: def 9
    ex i3 st i2 = i1 · i3;
 end;
```

The existence of two unrelated divisibility relations turned out to be an inconvenient feature of the current MML. Historically, natural numbers were introduced first and the theory of integers was developed later on the basis of naturals. However, at that time (1990) Mizar lacked expressive power to define one divisibility relation that could have been convenient in use for both naturals and integers. It should be expected that a future revision[4] of MML will leave only one divisibility relation and will introduce generalizations of theorems that at the moment are formulated for natural numbers.

In order to create an environment for rewriting Lamport's proof we have proved in the auxiliary file:

```
theorem :: LAUX_0:6
for n being Integer st 2 | n^2 holds 2 | n;
```

which is exactly the lemma assumed by Lamport. The proof easily follows from NAT_LAT:95. The problem of switching from natural divisibility to the integer one has been resolved by proving a technical fact:

```
theorem :: LAUX_0:5
for i1, i2 being Integer, n1, n2 being Nat st i1 = n1 & i2 = n2
  holds i1 | i2 iff n1 | n2;
```

[4]It seems that revisions and the resulting evolution of MML are unavoidable. The effort in maintaining an evolving data base of this type is not yet recognized as a substantial challenge.

2.6 Proof checking

Lamport's proof, although neatly formatted, is still an informal proof and some details are never mentioned. In Mizar we had to state all those details explicitly.

- In manipulating formulae involving division, it is common not to mention that the divisor is non zero when it can be inferred from some place in the proof. The fact that the variable n introduced in step $\langle 1\rangle 1$ and the term $gcd(p, q)$ are both not zero is not stated explicitly anywhere although both are used as divisors. In Mizar, we state these facts and have to refer to them (in step `1'1` labels `2'1a` and `2'1b`). Similarly, we have to state and refer to the fact that n^2 is non-zero (label `1'1_3a`).

- In some places the author makes an explicit reference to the definition of gcd (step $\langle 1\rangle 1$, substep $\langle 2\rangle 4$, substep $\langle 3\rangle 3$) and sometimes he does not (step $\langle 1\rangle 1$, substep $\langle 2\rangle 2$). Also, there are no references to the definition of divisibility or the definition of $\mathbf{Q}$, although the properties of both are used.

- In step $\langle 1\rangle 1$, substep $\langle 2\rangle 4$ of the Lamport's proof we read:

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

  which left us puzzled for a while. s is introduced without saying where is it coming from. If s is a natural number then the proof may succeed. If s were to be considered as an integer than we cannot prove that $s = 1$ as s also can be equal -1. The latter must have been Lamport's intention as in the revised version of his note [9] the troubling line appears as:

  Prove: $s = \pm 1$

  The original text was also published in [10][5].

- In step $\langle 1\rangle 2$, substep $\langle 2\rangle 1$ we see:

[5] In this context, the name of the publisher is worth noticing.

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

$\langle 2\rangle 1.$ $m^2 = 2n^2$

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

Probably a typographical error. $(m/n)^2 = 2$ follows from $\langle 1\rangle 1.2$ and $\langle 0\rangle{:}2$, which are used in step $\langle 1\rangle 3$ substep $\langle 2\rangle 2$ where the same fact is justified for the second time.

3 Conclusions

Formatting proofs can substantially help in reading proofs or even in writing them. However, proof formatting is a subjective matter and can not guarantee correctness of the formatted proof. In order to assure correctness of proofs we need to employ mechanical proof checkers. It particularly applies to "... correctness proofs of algorithms. These proofs are seldom deep, but usually have considerable detail." ([8], p. 9).

We have been working on such a mechanical checker for many years. The three main components of this system are: (1) a language for recording proofs, (2) software for processing the language and checking logical correctness, and (3) a growing data base of mathematical facts. All these three components evolve and our guess is that they will evolve for a long time before they stabilize.

Our experiment in rewriting Lamport's proof revealed a few inconveniences in the organization of the MIZAR data base. This is not an isolated case: the evolution of the data base can only be driven by using it extensively and only then such inconveniences can be discovered. The success of automated proof-checking in practice of mathematical reasoning heavily depends on the existence of a suitable data base of already checked mathematical facts. Building such a data base is a challenging task requiring the effort of many.

References

[1] Grzegorz Bancerek. The fundamental properties of natural numbers. *Formalized Mathematics*, 1(**1**):41–46, 1990.

[2] R. S. Boyer. A Mechanically Proof-Checked Encyclopedia of Mathematics: Should We Build One? Can We? In A. Bundy, editor, *12th International Conference on Automated Deduction LNAI/LNCS Vol. 814*, pages 237–251. Springer-Verlag, 1994.

[3] Marek Chmur. The lattice of natural numbers and the sublattice of it. The set of prime numbers. *Formalized Mathematics*, 2(**4**):453–459, 1991.

[4] J. Harrison. Reflections on Formalized Mathematics (DRAFT). `http://www.cl.cam.ac.uk/users/jrh/papers/form-math.html`, 1995.

[5] Krzysztof Hryniewiecki. Basic properties of real numbers. *Formalized Mathematics*, 1(**1**):35–40, 1990.

[6] Andrzej Kondracki. Basic properties of rational numbers. *Formalized Mathematics*, 1(**5**):841–845, 1990.

[7] Rafał Kwiatek and Grzegorz Zwara. The divisibility of integers and integer relative primes. *Formalized Mathematics*, 1(**5**):829–832, 1990.

[8] L. Lamport. How to Write a Proof. Research Report 94, DEC Systems Reserach Center, Palo Alto, CA, 1993.

[9] L. Lamport. How to Write a Proof. `http://www.research.digital.com/SRC/proofs/proofs.html`, 1993.

[10] L. Lamport. How to Write a Proof. In *Global Analysis in Modern Mathematics*, pages 311–321. Publish or Perish, Inc., 1993.

[11] P. Rudnicki. An Overview of the MIZAR Project. In *Proceedings of 1992 Workshop on Types and Proofs for Programs*, pages 311–332, June 1992. Available through anonymous ftp: `pub/cs-reports/Bastad92/proc.ps.Z` on `ftp.cs.chalmers.se`.

[12] Andrzej Trybulec and Czesław Byliński. Some properties of real numbers. *Formalized Mathematics*, 1(**3**):445–449, 1990.

[13] Michał J. Trybulec. Integers. *Formalized Mathematics*, 1(**3**):501–505, 1990.

A Our proof

Here is our proof of the irrationality of $\sqrt{2}$. The part of MIZAR text between environ and begin lists the data base units from which the processor imports symbols, notations, terminology, definitions, and theorems.

```
environ
 vocabulary   REAL_1, SQUARE, RAT_1, NAT_1;
 constructors REAL_1, SQUARE_1, RAT_1, INT_2, NAT_1;
 notation     ARYTM, REAL_1, SQUARE_1, RAT_1, NAT_1;
 theorems     AXIOMS, REAL_1, NAT_1, SQUARE_1, INT_1, INT_2,
              RAT_1, NAT_LAT;
 clusters     INT_1, RAT_1;
 requirements ARYTM;
begin

Fact: not ex r being Rational st 0 ≤ r & r² = 2
proof given r being Rational such that
A0'1: 0 ≤ r and
A0'2: r² = 2;
   set m = numerator r, n = denominator r;
1'1: n <> 0                      by RAT_1:def 3; then
1'2: n·n <> 0                    by REAL_1:23;
       r = m/n                   by RAT_1:37; then
       (m·m)/(n·n) = r·r         by 1'1, REAL_1:35
                  .= 2           by A0'2, SQUARE_1:def 3; then
1'3: 2·(n·n) = m·m               by 1'2, REAL_1:43;
       0 ≤ m                     by A0'1, RAT_1:76; then
     reconsider m as Nat         by INT_1:16;
       2 | m·m                   by 1'3, NAT_1:def 3; then
       2 | m                     by INT_2:44, NAT_LAT:95; then
     consider m' being Nat such that
1'4: m = 2·m'                    by NAT_1:def 3;
 1'4a: 2 <> 0;
       m·m = 2·(m'·(m'·2))       by 1'4, AXIOMS:16; then
       n·n = m'·(m'·2)           by 1'3, 1'4a, REAL_1:12
          .= m'·m'·2             by AXIOMS:16; then
       2 | n·n                   by NAT_1:def 3; then
       2 | n                     by INT_2:44, NAT_LAT:95; then
     ex n' being Nat st n = 2·n' by NAT_1:def 3;
 hence contradiction             by 1'4, RAT_1:62;
end;

theorem not ex r being Rational st r² = 2
proof let r being Rational;
     (0 ≤ r or 0 ≤ -r) & r² = (-r)² by REAL_1:66, SQUARE_1:61;
   hence thesis by Fact;
end;
```

B Lamport's Proof

The following is quoted *in extenso* from [8].

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: $2 = (m/n)^2$ [$\langle 1 \rangle 1.2$ and $\langle 0 \rangle$:2]
    $= (2p/n)^2$ [$\langle 2 \rangle 1$]
    $= 4p^2/n^2$ [Algebra]
    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)

C Lamport's proof in Mizar

The following is Lamport's proof [8] written in Mizar. We tried not to depart from the original.

```
environ
 vocabulary   REAL_1, SQUARE, RAT_1, NAT_1, INT_1, INT_2D;
 constructors REAL_1, SQUARE_1, RAT_1, INT_2;
 notation     ARYTM, REAL_1, SQUARE_1, RAT_1, INT_1, INT_2;
 theorems     AXIOMS, REAL_1, SQUARE_1, INT_1, INT_2, RAT_1, REAL_2,
              LAUX_0;
 clusters     INT_1, LAUX_0;
 requirements ARYTM;
begin

theorem not ex r being Real st r ∈ RAT & r^2 = 2
proof given r being Real such that
A0'1: r ∈ RAT and
A0'2: r^2 = 2;

1'1: ex m, n being Integer st m gcd n = 1 & r = m/n & n<>0
     proof
          consider p, q being Integer such that
     2'1: q <> 0 & r = p/q                by A0'1, RAT_1:def 1;
     set  m = p/(p gcd q),
          n = q/(p gcd q);
        2'1a: p gcd q <> 0                by 2'1, INT_2:35;
        2'1b: n <> 0                      by 2'1, 2'1a, REAL_2:42;
     2'2: m ∈ INT & n ∈ INT               by 2'1, LAUX_0:1;
          reconsider m, n as Integer      by 2'2, INT_1:def 2;
     2'3: r = m/n                         by 2'1, 2'1a, REAL_2:55;
     2'4: m gcd n = 1
          proof
            now let s be Integer such that
            A2'1: s | m and
            A2'2: s | n;
            thus  s = 1 or s = -1
                  proof
                  3'1: s·(p gcd q) | p    by A2'1, 2'1a, LAUX_0:2;
                  3'2: s·(p gcd q) | q    by A2'2, 2'1a, LAUX_0:2;
                  thus thesis       by 3'1, 3'2, 2'1a, LAUX_0:4;
                  end;
            end;
          hence thesis              by LAUX_0:3;
          end;
     thus thesis                    by 2'4, 2'3, 2'1b;
     end;
```

```
     consider m, n being Integer such that
1'1_1: m gcd n = 1      and
1'1_2: r = m/n          and
1'1_3: n <> 0           by 1'1;

     1'1_3a: n^2 <> 0    by 1'1_3, SQUARE_1:73;

1'2: 2 | m
     proof
          m^2/n^2 = (m/n)^2       by 1'1_3, SQUARE_1:69; then
     2'1: m^2 = 2·n^2            by 1'1_3a, 1'1_2, A0'2, REAL_1:43;
          2 | m^2                by 2'1, INT_1:def 9;
     hence thesis                by LAUX_0:6;
     end;
1'3: 2 | n
     proof
          consider p being Integer such that
     2'1: m = 2·p                       by 1'2, INT_1:def 9;
     2'2: n^2 = 2·p^2
          proof
          3'1: 2·1 = (m/n)^2            by 1'1_2, A0'2
                  .= (2·p/n)^2          by 2'1
                  .= (2·p)^2/n^2        by 1'1_3, SQUARE_1:69
                  .= 2^2·p^2/n^2        by SQUARE_1:68
                  .= 2^2·(p^2/n^2)      by 1'1_3a, SQUARE_1:18
                  .= 2·2·(p^2/n^2)      by SQUARE_1:def 3
                  .= 2·(2·(p^2/n^2))    by AXIOMS:16;
               2 <> 0; then
          3'2: 1 = 2·(p^2/n^2)          by 3'1, REAL_1:12
                .= 2·p^2/n^2            by 1'1_3a, SQUARE_1:18;
          thus thesis                   by 3'2, 1'1_3a, REAL_2:30;
          end;
          2 | n^2       by 2'2, INT_1:def 9;
     hence thesis       by LAUX_0:6;
     end;

     2 | 1              by 1'2, 1'3, 1'1_1, INT_2:33; then
     2 = 1 or 2 = -1    by INT_2:17;
hence thesis            by INT_2:9;
end;
```

D Abstract of the auxiliary file

The statements of the auxiliary facts (MIZAR file LAUX_0), the length of the file with proofs was under 140 lines.

```
environ
 vocabulary   REAL_1, SQUARE, NAT_1, INT_1, INT_2D;
 constructors REAL_1, SQUARE_1, NAT_1, INT_2;
 notation     ARYTM, REAL_1, SQUARE_1, INT_1, NAT_1, INT_2;
 theorems     AXIOMS, REAL_1, NAT_1, SQUARE_1, INT_1, INT_2,
              REAL_2, NAT_LAT;
 clusters     INT_1;
 definitions  NAT_1, INT_1;
 requirements ARYTM;
begin

definition let i be Integer;
 cluster i^2 -> integer;
end;

theorem :: LAUX_0:1
for p, q being Integer st q <> 0
  holds p/(p gcd q) ∈ INT & q/(p gcd q) ∈ INT;

theorem :: LAUX_0:2
for m, p, d, s being Integer st m = p/d & s | m & d<>0
  holds s·d | p;

theorem :: LAUX_0:3
for m, n being Integer
 st for s being Integer st s | m & s | n holds s = 1 or s = -1
  holds m gcd n = 1;

theorem :: LAUX_0:4
for s, p, q being Integer
 st s·(p gcd q) | p & s·(p gcd q) | q & p gcd q <> 0
  holds s = 1 or s = -1;

theorem :: LAUX_0:5
for i1, i2 being Integer, n1, n2 being Nat st i1 = n1 & i2 = n2
  holds i1 | i2 iff n1 | n2;

theorem :: LAUX_0:6
for n being Integer st 2 | n^2 holds 2 | n;
```