Why We Need Structured Proofs in Mathematics

Mauricio Ayala-Rincón
ayala@unb.br

Gabriel Ferreira Silva
gabrielfsilva1995@gmail.com

Departments of Computer Science and Mathematics, Universidade de Brasília

When mathematicians write a standard proof of some theorem, they have to decide on which level of detail they will present their argument. If they provide too little justification, readers may spend a lot of time filling the holes left or, even worse, may not understand why a specific step of the proof is (or is not) correct. However, more argumentation is not necessarily always better, since this may obscure the "big picture" and some readers may be more interested in seeing the "big picture" than in checking every tiny detail of the proof. Therefore, the ideal level of detail in a mathematical proof varies from reader to reader, as it depends on the reader's previous knowledge and the reader's intention.

The standard way of writing mathematical proofs, which will henceforth be called prose proofs, cannot satisfy everyone. In contrast with prose proofs, we have the concept of structured proofs, proposed by Leslie Lamport in [8] and described in [9, 8] and also in [10]. In a structured proof, the steps and substeps are hierarchically presented. The proof is decomposed as a tree of steps, and each step can be further decomposed into smaller subtrees of substeps and so on. By correctly numerating each step and by using the right indentation, the hierarchy of the proof is clear at first sight. Examples of structured proofs can be found in Appendix 1 and Appendix 2.

If writing structured proofs is accompanied by the discipline of explaining every step in meticulous detail, the probability of obtaining an incorrect proof or a wrong result diminishes. That is because the hierarchy of the proof will allow the writer to add justifications to a given step without "clouding" other steps of the proof, and also make it easier to check if all the corner cases were handled.

Since structured proofs can make it harder to obtain an incorrect result, a comparison may arise between structured proofs and interactive theorem provers (ITPs). By using an ITP, the chances of obtaining an incorrect result are significantly lower than by using a structured proof, as the computer checks every step of the proof. However, structured proofs also have advantages over ITPs, as they are faster to write and more readable. As members of a research group bridging interest from graduate programs in Mathematics and Computer Science we also have a huge interest in motivating mathematicians to use ITPs. For this we have developed mathematical theories in PVS as well as short-courses to attract the interest of such audience [5, 3].

Remark 1. The LaTeX package `pf2` (see [1]) can be used to help writing structured proofs in mathematical papers.

Remark 2. As hinted in the text, a structured proof can be seen as a tree. Indeed, in proof theory it is well-known that the structure of a proof is a tree. For instance, derivations using Gentzen's calculus for first-order logic correspond to trees that coincide also with the tree structure of formula (by inspection on the rules of the Gentzen system, see Definition 3.1.2 in [11]) With this view in mind, note that:

- The main steps of the proof correspond to the immediate childs of the root.
- When we move away from the main steps of the structured proof to go to the tiny details that make the proof work, this corresponds to "going deeper" in the tree of the structured proof, traveling branches and getting farther from the root.

References

[1] `http://lamport.azurewebsites.net/latex/latex.html`. Accessed at July 8, 2020.

In: A. Editor, B. Coeditor (eds.): Proceedings of the XYZ Workshop, Location, Country, DD-MMM-YYYY, published at http://ceur-ws.org

[2] https://math.stackexchange.com/questions/858978/lamport-claims-there-is-an-error-in-kelleys-proof-of-the-schroeder-bernstein-th. Accessed at July 6, 2020.

[3] Mauricio Ayala-Rincón and Thaynara Arielly de Lima. Teaching Interactive Proofs to Mathematicians. In *Post-proceedings 9th International Workshop on Theorem Prover Components for Educational Software (ThEdu). In press EPTCS*, 2020.

[4] Choukri-Bey Ben-Yelles. *Type assignment in the lambda-calculus: Syntax and semantics.* PhD thesis, University College of Swansea, 1979.

[5] Thaynara Arielly de Lima, André Luiz Galdino, Andréia B. Avelar, and Mauricio Ayala-Rincón. Formalization of Ring Theory in PVS - Isomorphism Theorems, Principal, Prime and Maximal Ideals, Chinese Remainder Theorem. Technical report, 2020. http://ayala.mat.unb.br/publications.html.

[6] J Roger Hindley. *Basic simple type theory.* Cambridge University Press, 1997.

[7] John L Kelley. *General topology.* 1975.

[8] Leslie Lamport. How to write a proof. *The American math. monthly*, 102(7):600–608, 1995.

[9] Leslie Lamport. How to write a 21st century proof. *J. of Fixed Point Theory and Applications*, 11(1):43–63, 2012.

[10] Gabriel Ferreira Silva. Why We Need Structured Proofs, 2020. Available at https://medium.com/@gabrielferreirasilva/why-we-need-structured-proofs-in-mathematics-34a3034f2f90.

[11] Anne Sjerp Troelstra and Helmut Schwichtenberg. *Basic proof theory.* Number 43. Cambridge University Press, 2000.

1 Structured Proofs: An Example From Type Theory

In this section we present a structured proof using an example from type theory. Since this example comes from a seminar one of the authors presented, we also report our thoughts on the presentation and the opinion of the audience.

The differences between prose proofs and structured proofs are less noticeable in short proofs. Therefore, we opted to present a theorem whose proof is long and relies on previous definitions and lemmas to make it clear how the hierarchical organization let us immediately distinguish between the main steps and the detailed justification for each step and substep.

1.1 The Theorem To Be Proved

The question of counting a type's inhabitant is central in type theory [6]. It originated with the work of Ben-Yelles in 1979 (see [4]) where an algorithm for the task was proposed. The counting algorithm (8D5 in [6]) takes a type τ and outputs the number of its normal inhabitants (modulo change of bound variables) and then list these one by one, deciding in finite time whether the list will be finite or not. The next theorem is used in the proof of the correctness of the counting algorithm. It is Theorem 8F3 of Hindley's book "Basic Simple Type Theory" [6]:

Lemma 1 (8F3 in Hindley's book (see [6]))**.** If $\mathit{Long}(\tau)$ has a member M^τ with depth $\geq \mathbb{D}(\tau)$ then

1. it has a member $M^{*\tau}$ with
$$\mathit{Depth}(M^\tau) - ||\tau|| \leq \mathit{Depth}(M^{*\tau}) < \mathit{Depth}(M^\tau)$$

2. it has a member N^τ with
$$\mathbb{D}(\tau) - ||\tau|| \leq \mathit{Depth}(N^\tau) < \mathbb{D}(\tau)$$

1.2 Our Structured Proof

Our structured proof is presented below. The notation of the steps and substeps follows [9]. We say that a step has depth 1 if it is used in the proof of the lemma. We say that a substep has depth $n+1$ if it is used in the proof of a step of depth n. Therefore, a substep numbered as $\langle 3\rangle 5$, for instance, has depth 3 and is the 5^{th} substep that compose the proof of a step of depth $3-1=2$.

$\langle 1\rangle 1$. If $\mathit{Long}(\tau)$ has a member M^τ with depth $\geq \mathbb{D}(\tau)$ then it has a member $M^{*\tau}$ with:
$$\mathit{Depth}(M^\tau) - ||\tau|| \leq \mathit{Depth}(M^{*\tau}) < \mathit{Depth}(M^\tau)$$

$\langle 2\rangle 1$. Let: M be a member of $\mathit{Long}(\tau)$ without bound-variable clashes.

$\langle 2\rangle 2$. Let: $d = \mathit{Depth}(M)$. $d \geq \mathbb{D}(\tau) \geq 2$.
$\langle 3\rangle 1$. $d = \mathit{Depth}(M) > \mathbb{D}(\tau)$ by hypothesis.
$\langle 3\rangle 2$. By Definition, $\mathbb{D}(\tau) = |\tau| \times ||\tau||$.
$\langle 3\rangle 3$. $|\tau| \geq 2$ since τ is composite. Notice that τ must be composite since atomic types have no inhabitants.
$\langle 3\rangle 4$. $\mathbb{D}(\tau) \geq 2$.

$\langle 2\rangle 3$. Consider any argument-branch of M with lenght d. It has form
$$\langle N_0, \ldots, N_d\rangle$$
where $\underline{\mathrm{N}}_0 \equiv \underline{\mathrm{M}}$ and $\underline{\mathrm{N}}_{i+1}$ is an argument of $\underline{\mathrm{N}}_i$ for $i = 0, \ldots, d-1$. We will shrink this branch.
By 8E4.1, since $\mathit{Depth}(M) = d$, M has at least one argument-branch with length d.

$\langle 2\rangle 4$. Each N_i has form
$$N_i \equiv \lambda x_{i,1} \ldots x_{i,m_i}.y_i P_{i,1} \ldots P_{i,n_i} \qquad (m_i, n_i \geq 0)$$

$\langle 2\rangle 5$. Let: $\rho_i \equiv \rho_{i,1} \to \ldots \to \rho_{i,m_i} \to a_i$ be the type of N_i.

$\langle 2\rangle 6$. $IAT(N_i) = \langle \rho_{i,1}, \ldots, \rho_{i,m_i}\rangle$.
Since $\underline{\mathrm{N}}_i$ is long, the types of $x_{i,1}, x_{i,2}, \ldots$ are exactly $\rho_{i,1}, \rho_{i,2}, \ldots$. By the definition of IAT we obtain $IAT(N_i) = \langle \rho_{i,1}, \ldots, \rho_{i,m_i}\rangle$.

⟨2⟩7. LET: $\underline{\mathrm{B}}_i$ be the body of N_i, just as in the proof of Lemma 8F2 (the previous lemma). The type of $\underline{\mathrm{B}}_i$ is a_i.

Since the type of $\underline{\mathrm{B}}_i$ is the tail of the type of $\underline{\mathrm{N}}_i$.

⟨2⟩8. LET: the sequence $d_0, d_1, \ldots$ be defined as follows. $d_0 = 0$. d_{j+1} is the least index greater than d_j such that $IAT(N_{d_{j+1}})$ differs from all of:

$$IAT(N_{d_0}), \ldots, IAT(N_{d_j})$$

.

⟨2⟩9. LET: n be the greatest integer such that d_n is defined.

⟨2⟩10. $d_0, \ldots, d_n$ partition the set $\{0, 1, \ldots, d\}$ into the following $n + 1$, non empty sets, which will be called **IAT-intervals**:

$$\mathbb{I}_j = \{d_j, d_j + 1, \ldots, d_{j+1} - 1\} \qquad (0 \leq j \leq n - 1)$$
$$\mathbb{I}_n = \{d_n, d_n + 1, \ldots, d\}$$

⟨2⟩11. If $\mathbb{I}_j$ contains two numbers p and $p + r$, with $r \geq 1$ and B_p and B_{p+r} have the same type we shal call $\langle p, p + r\rangle$ a **tail-repetition**. It will be called **minimal** iff there is no other tail-repetition $\langle p', q'\rangle$ with $p \leq p' < q' \leq p + r$.

⟨2⟩12. At least one IAT-interval contains a tail-repetition.

⟨3⟩1. Suppose, by contradiction, that no interval contained a tail-repetition.

⟨3⟩2. An $\mathbb{I}_j$ that contains no tail-repetition must have $\leq ||\tau||$ members.

⟨4⟩1. For such an $\mathbb{I}_j$, the atoms:

$$a_{d_j}, \ldots, a_{d_{j+1}-1}$$

must all be distinct.

⟨4⟩2. By Step ⟨2⟩5, each a_i occurs in ρ_i.

⟨4⟩3. By 8E7, ρ_i occurs in τ. So, a_i occurs in τ.

⟨4⟩4. By definition, there are only $||\tau||$ distinct atoms in τ.

⟨4⟩5. Hence, $\mathbb{I}_j$ has $\leq ||\tau||$ members.

⟨3⟩3. Since there are $n + 1$ IAT intervals in the given branch, the branch would have $\leq (n + 1) \times ||\tau||$ members.

⟨3⟩4. $n + 1 \leq |\tau|$. So, the branch would have $\leq |\tau| \times ||\tau||$ members.

⟨4⟩1. Since our argument-branch has d members after $\underline{\mathrm{N}}_0$, we have $n \leq d$ and $d_n \leq d$.

⟨4⟩2. $0 = d_0 < d_1 < \ldots < d_n \leq d$.

⟨4⟩3. For each i, $IAT(N_i)$ is identical to one of:

$$IAT(N_{d_0}), IAT(N_{d_1}), \ldots, IAT(N_{d_n})$$

where each one of the IAT's in the equation above are distinct.

⟨4⟩4. $n + 1 \leq \#(NSS(\tau)) + 1$

By 8E7, each one of the $n + 1$ $IAT's$ are empty or members of $NSS(\tau)$. Since they are distinct, at most one of them is empty.

⟨4⟩5. $\#(NSS(\tau)) \leq |\tau| - 1$

By 9E9.3(ii)

⟨3⟩5. However the branch has $d + 1$ members and using Step ⟨2⟩2 we obtain

$$d + 1 = Depth(M) + 1 \geq \mathbb{D}(\tau) + 1 > |\tau| \times ||\tau||$$

which contradicts Step ⟨3⟩4.

⟨2⟩13. We start to build M^* as follows. In the given branch take the last $\mathbb{I}_j$ containing a tail-repetition, choose a minimal tail-repetition $\langle p, p + r\rangle$ in it and change M to a new term M' by replacing B_p by B_{p+r}.

⟨2⟩14. M' is a genuine typed term. M' is a long β-nf with the same type as M. Also $|M'| < |M|$.

⟨3⟩1. M' is a genuine typed term, with the same type as M.

We repeat the argument used in the proof of the Stretching Lemma (8F2):

⟨4⟩1. Let: Γ_i be the context that assigns to the initial abstractors of N_i the types they have in M.

⟨4⟩2. The set $Con(B_{p+r}) \cup Con(M) \cup \Gamma_0 \cup \ldots \cup \Gamma_p$ is consistent.

⟨5⟩1. $\Gamma_0 \cup \ldots \cup \Gamma_d$ is consistent.
Since M has no bound variable clashes, the variables in $\Gamma_0, \ldots, \Gamma_d$ are all distinct.

⟨5⟩2. $Con(B_{p+r}) \subseteq \Gamma_0 \cup \ldots \cup \Gamma_d$.

⟨6⟩1. Every variable free in B_{p+r} is bound in one of $N_0, \ldots, N_{p+r}$ because M is closed and $\underline{\mathrm{B}}_{p+r}$ is in $\underline{\mathrm{N}}_{p+r}$.
⟨6⟩2. Therefore, by the definition of typed term (5A1) we get $B_{p+r} \in \mathbb{TT}(\Gamma_0 \cup \ldots \cup \Gamma_{p+r})$.
⟨6⟩3. By the definition of $Con()$ we obtain $Con(B_{p+r}) \subseteq \Gamma_0 \cup \ldots \cup \Gamma_{p+r}$.
⟨6⟩4. $\Gamma_0 \cup \ldots \cup \Gamma_{p+r} \subseteq \Gamma_0 \cup \ldots \cup \Gamma_d$.

⟨5⟩3. $Con(M) \subseteq \Gamma_0 \cup \ldots \cup \Gamma_d$.
Since M is closed, $Con(M) = \emptyset$.

⟨5⟩4. $\Gamma_0 \cup \ldots \cup \Gamma_{p+r} \subseteq \Gamma_0 \cup \ldots \cup \Gamma_d$.

⟨4⟩3. Since M is a genuine typed term and Step ⟨4⟩2 holds and the abstractors in M whose scope contain $\underline{\mathrm{B}}_p$, are exactly the initial abstractors of $N_0, \ldots, N_p$ we can apply Lemma 5B2.1(ii) and conclude that M' is a genuine typed term with the same type as M.

⟨3⟩2. M' is a long β-nf.
Since M is a long β-nf and B_p and B_{p+r} have the same type.

⟨3⟩3. $|M'| < |M|$.
Since B_p properly contains B_{p+r} we have $|B_{p+r}| < |B_p|$ and hence $|M'| < |M|$.

⟨2⟩15. Although M' might not be closed, there is a procedure in which, from M', we can obtain a long β-nf M'' with the same type and depth as M' which is closed. Notice that we are not claiming that M' and M'' are related by α-conversion or any other way.

⟨3⟩1. First, notice that M' might not be closed.
M' might not be closed because the change from M to M' has removed the initial abstractors of $\underline{\mathrm{N}}_{p+1}, \ldots, \underline{\mathrm{N}}_{p+r}$ from M, and so some free variables occurrences in $\underline{\mathrm{B}}_{p+r}$ that were bound in M might now be free in M'.

⟨3⟩2. Let: $\underline{\mathrm{v}}$ be free in the occurrence of $\underline{\mathrm{B}}_{p+r}$ in M' that has replaced $\underline{\mathrm{B}}_p$ in M. Let: $\underline{\mathrm{v}}$ be also free in M'.

⟨3⟩3. There is a variable in $x_{d_q,k} \in IA(N_{d_q})$, with $d_q \leq p$ that has the same type as v.

⟨4⟩1. v occurs in $IA(\underline{\mathrm{N}}_h)$ for some h with $p+1 \leq h \leq p+r$.
Since $\underline{\mathrm{v}}$ is free in M', v does not occur in a covering abstractor of this occurrence of B_{p+r} in M'. This covering abstractors are exactly the initial abstractors of $\underline{\mathrm{N}}_0, \ldots, \underline{\mathrm{N}}_p$ in M so:

$$v \notin IA(\underline{\mathrm{N}}_0) \cup \ldots \cup IA(\underline{\mathrm{N}}_p)$$

However, M is closed and therefore our $\underline{\mathrm{v}}$, in M, must be in the scope of a $\underline{\lambda v}$ in one of $IA(\underline{\mathrm{N}}_0), \ldots, IA(\underline{\mathrm{N}}_{p+r})$. Hence, v occurs in $IA(\underline{\mathrm{N}}_h)$ for some h with $p+1 \leq h \leq p+r$.

⟨4⟩2. In our notation, we have $v \equiv x_{h,k}$ for some $k \leq m_h$. Also, the type of v is $\rho_{h,k} \in IAT(\underline{\mathrm{N}}_h)$.

⟨4⟩3. $IAT(\underline{\mathrm{N}}_h) = IAT(\underline{\mathrm{N}}_{d_q})$ for some $q \leq j$.
Since the tail-repetition $\langle p, p+r \rangle$ is in the interval $\mathbb{I}_j$, by our definition of $d_0, \ldots, d_n$, we get that $IAT(\underline{\mathrm{N}}_h)$ coincides with:

$$IAT(\underline{\mathrm{N}}_{d_0}), \ldots, IAT(\underline{\mathrm{N}}_{d_j})$$

⟨4⟩4. Hence, there is a variable $x_{d_q,k} \in IA(N_{d_q})$ with the same type as v.

⟨4⟩5. $d_q \leq p$.

From Step ⟨4⟩3, we have $q \leq j$, which implies $d_q \leq d_j$. Since the tail-repetition $\langle p, p+r \rangle$ occurs in $\mathbb{I}_j$ we have $p \geq d_j$.

⟨3⟩4. Replace v by this variable. The result will be a long β-nf with the same type and depth as M' and containing one less free variable.

From ⟨3⟩3, we see that this variable is bound by an abstractor in N_{d_q}, where $d_q \leq p$. Since the change from M to M' has only removed the initial abstractors of $\underline{\mathrm{N}}_{p+1}, \ldots, \underline{\mathrm{N}}_{p+r}$, this variable is still a bound variable in M'. Therefore, the result has one less free variable than M'. The result has the same type and depth because we substituted a variable v by another variable that has the same type as v.

⟨3⟩5. By similarly replacing every variable of $\underline{\mathrm{B}}_{p+r}$ that is free in M' by a new one which has the same type but is bound in M' we obtain a long β-nf M'' with the same type and depth as M' and which is closed.

⟨2⟩16. $d - ||\tau|| \leq \mathit{Depth}(M'') \leq d.$

⟨3⟩1. The number of arguments removed from the argument-branch is r, so our argument-branch now contains $d - r$ arguments.

⟨3⟩2. Hence, $d - r \leq \mathit{Depth}(M'') \leq d.$

⟨3⟩3. $r \leq ||\tau||.$

By definition, there are only $||\tau||$ distinct atoms in τ. Since the tail repetition $\langle p, p+r \rangle$ we took is minimal, we have $r \leq ||\tau||$.

⟨3⟩4. $d - ||\tau|| \leq \mathit{Depth}(M'') \leq d.$

⟨2⟩17. If $\mathit{Depth}(M'') < d$ define $M^* \equiv M''$. If not, select a branch in M'' with length d and apply the removal procedure to it (the removal procedure is the one that from M produced M''). Keep doing this to shorten the branches with length d until there are none left. Define M^* to be the first term produced by this procedure whose depth is less than d.

⟨2⟩18. Then:

$$d - ||\tau|| \leq \mathit{Depth}(M^*) < d$$

as required.

⟨1⟩2. If $\mathit{Long}(\tau)$ has a member M^τ with depth $\geq \mathbb{D}(\tau)$ then it has a member N^τ with:

$$\mathbb{D}(\tau) - ||\tau|| \leq \mathit{Depth}(N^\tau) < \mathbb{D}(\tau)$$

By repeating the whole procedure described in Step ⟨1⟩1 until you obtain an output with depth $< \mathbb{D}(\tau)$.

1.3 Comments

The audience liked the presentation of this proof in a structured manner, commenting that the indentation helped distinguish the main steps from the substeps and that the talk “flowed” well. Advantage was taken from the hierarchical organization of the proof when explaining the lengthy proof of the first item of the lemma (Step ⟨1⟩1). First, all the immediate substeps of Step ⟨1⟩1, i.e., the substeps with depth 2, were explained. This gave the audience the “big picture” and the intuition for why the proof works. Since there was no time left, the speaker could not get into the details of each substep, but made the slides of the talk, which contained every detail of the proof, available. This incident reveals yet another advantage of structured proofs over prose proofs: if the speaker giving a talk is running out of time, it is easier to only explain the main steps of the proof when the proof is organized in a hierarchical manner.

2 An Example of How Structured Proofs Makes Writing Wrong Proofs Harder

We mentioned that structured proofs, when combined with the discipline to provide detail for every step, makes writing wrong proofs harder than traditional prose proofs. We illustrate our case with an example. The idea for this example comes from [8]. In [8], Leslie Lamport also talks about his experience writing structured proofs:

> "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 proving 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."

We enunciate the Schroeder-Bernstein theorem in Section 2.1. Since [8] does not present the error in Kelley's (see [7], page 28) textbook, we present the wrong proof (pointing the incorrect step) in Section 2.2 and present a corrected version of the proof in Section 2.3.

Remark 3. The error in Kelley's proof is also explained in [2]. The structured version of the proof presented here is, to the best of our knowledge, new.

Remark 4. Kelley (see [7]) attributes the form of the proof to G.Birkhoff and S. MacLane:

> "The intuitively elegant form of the proof of theorem 0.20 is due to G.Birkhoff and S. MacLane".

2.1 The Theorem To Be Proved

The Schroeder-Bernstein theorem uses the definition of equipollent sets, which we present now.

Definition 1 (Equipollent Set)**.** Two sets A and B are said to be equipollent iff there is a one-to-one function on A with range B.

With this concept clear, we now state the Schroeder-Bernstein theorem:

Theorem 1 (Schroeder-Bernstein Theorem)**.** If there is a one-to-one function on a set A to a subset of a set B and there is also a one-to-one function on B to a subset of A, then A and B are equipollent.

2.2 A Wrong Prose Proof

2.2.1 The Prose Proof

This prose proof is from [7], starting at page 28:

> "PROOF Suppose that f is a one-to-one map of A into B and g is one-to-one on B to A. It may be supposed that A and B are distinct. The proof of the theorem is accomplished by decomposing A and B into classes which are most easily described in terms of parthenogenesis. A point x (of either A or B) is an ancestor of a point y iff y can be obtained from x by successive application of f and g (or g and f). Now decompose A into three sets: let A_E consist of all points of A which have an even number of ancestors, let A_O consists of points which have an odd number of ancestors, and let A_I consist of points with infinitely many ancestors. Decompose B similarly and observe: f maps A_E onto B_O and A_I onto B_I, and g^{-1} maps A_O onto B_E. Hence, the function which agrees with f on $A_E \cup A_I$ and agrees with g^{-1} on A_O is a one-to-one map of A onto B. □"

2.2.2 Why It's Wrong

The error in the proof is in the affirmation:

> "f maps A_E onto B_O"

since it does not take into account the possibility of cycles. For instance, if we have a such that its ancestors are $f(a)$ and $g(f(a)) = a$, the ancestors of $f(a)$ are also just a and $f(a)$. In this case, we had $a \in A_E$ and $f(a) \in B_E$, contradicting the claim.

There is a similar error in the affirmation:

> "g^{-1} maps A_O onto B_E."

2.3 A Correct Structured Proof

To fix Kelley's proof, we defined the ancestors of an element as a sequence instead of a set. Here is a structured proof of the Schroeder-Bernstein Theorem:

$\langle 1 \rangle 1$. LET: f be a one-to-one map of A into B and g be a one-to-one map of B into A.

$\langle 1 \rangle 2$. CASE: A and B are disjoint.

$\langle 2 \rangle 1$. Since f and g are one-to-one we can use without ambiguity the mapping f^{-1} for elements $b \in f(A) \subseteq B$ and the mapping g^{-1} for elements $a \in g(B) \subseteq A$.

$\langle 2 \rangle 2$. LET: $a \in A$. DEFINE: $x_0 = a$ as the zeroth ancestor of a. If $x_0 \in g(B)$, DEFINE: $x_1 = g^{-1}(x_0)$ as the first ancestor of a. If $x_0 \notin g(B)$, the sequence of ancestors of a is just $\langle x_0 \rangle$. If $x_1 \in f(A)$, DEFINE: $x_2 = f^{-1}(x_1)$ as the second ancestor of a. If $x_1 \notin f(A)$, the sequence of ancestors of a is just $\langle x_0, x_1 \rangle$. DEFINE: $Anc(a)$ to be the sequence (possibly infinite) of ancestors of a obtained by continuing this process for as long as we can. DEFINE: $|Anc(a)|$ as the number of elements (including x_0) in $Anc(a)$.

$\langle 2 \rangle 3$. We adapt Step $\langle 2 \rangle 2$ to an element $b \in B$. DEFINE: $x_0 = b$ as the zeroth ancestor of b. If $x_0 \in f(A)$, DEFINE: $x_1 = f^{-1}(x_0)$ as the first ancestor of b. If $x_0 \notin f(A)$, the sequence of ancestors of b is just $\langle x_0 \rangle$. If $x_1 \in g(B)$, DEFINE: $x_2 = g^{-1}(x_1)$ as the second ancestor of b. If $x_1 \notin g(B)$, the sequence of ancestors of b is just $\langle x_0, x_1 \rangle$. DEFINE: $Anc(b)$ to be the sequence (possibly infinite) of ancestors of b obtained by continuing this process for as long as we can. DEFINE: $|Anc(b)|$ as the number of elements (including x_0) in $Anc(b)$. Since A and B are disjoint (see Step $\langle 1 \rangle 2$), there is no danger of Step $\langle 2 \rangle 2$ and this Step simultaneously defininig, for some $x \in A \cup B$, $Anc(x)$.

$\langle 2 \rangle 4$. LET: $A_E = \{a \mid a \in A \text{ and } |Anc(a)| \text{ is even }\}$. LET: $A_O = \{a \mid a \in A \text{ and } |Anc(a)| \text{ is odd }\}$. LET: $A_I = \{a \mid a \in A \text{ and } |Anc(a)| = \infty\}$. We have $A = A_E \uplus A_O \uplus A_I$.

$\langle 2 \rangle 5$. Similar to Step $\langle 2 \rangle 4$, we partition B conveniently. LET: $B_E = \{b \mid b \in B \text{ and } |Anc(b)| \text{ is even }\}$. LET: $B_O = \{b \mid b \in B \text{ and } |Anc(b)| \text{ is odd }\}$. LET: $B_I = \{b \mid b \in B \text{ and } |Anc(b)| = \infty\}$. We have $B = B_E \uplus B_O \uplus B_I$.

$\langle 2 \rangle 6$. f maps A_I onto B_I and A_O onto B_E. g^{-1} maps A_E onto B_O.

$\langle 3 \rangle 1$. f maps A_I onto B_I.

$\langle 4 \rangle 1$. LET: $a \in A_I$. Then, $Anc(a) = \langle x_0 = a, x_1, \ldots \rangle$ is an infinite sequence. By definition, $Anc(f(a)) = \langle f(a), f^{-1}(f(a)) = a = x_0, x_1, \ldots \rangle$, which is also an infinite sequence. Therefore we have $f(a) \in B_I$. We conclude that f maps A_I in B_I.

$\langle 4 \rangle 2$. LET: $b \in B_I$. Then, $Anc(b) = \langle x_0 = b, x_1 = f^{-1}(b), x_2, \ldots \rangle$ is an infinite sequence. PICK $a = f^{-1}(b)$. We have $Anc(a) = \langle a = f^{-1}(b) = x_1, x_2, \ldots \rangle$, which is also an infinite sequence. Therefore, $a \in A_I$ with $f(a) = b$. We conclude that the function f maps A_I onto B_I.

$\langle 3 \rangle 2$. f maps A_O onto B_E.

$\langle 4 \rangle 1$. LET: $a \in A_O$. Then, $|Anc(a)| = 2k + 1$, with $k \in \mathbb{N}$ and $Anc(a)$ is of the form: $\langle x_0 = a, \ldots, x_{2k} \rangle$. By definition, $Anc(f(a)) = \langle f(a), f^{-1}(f(a)) = x_0 = a, \ldots, x_{2k} \rangle$ and we have $|Anc(f(a))| = 2k + 2$. Hence, $f(a) \in B_E$ and we conclude that f maps A_O in B_E.

$\langle 4 \rangle 2$. LET: $b \in B_E$. Then $|Anc(b)| = 2k$, with $k \in \mathbb{N}^*$ and $Anc(b)$ is of the form:
$\langle x_0 = b, x_1 = f^{-1}(b), \ldots, x_{2k-1} \rangle$. PICK $a = f^{-1}(b)$. Then, $Anc(a) = \langle x_1 = a = f^{-1}(b), \ldots, x_{2k-1} \rangle$, $|Anc(a)| = 2k - 1$ and hence $a \in A_O$ with $f(a) = b$. We conclude that f maps A_O onto B_E.

$\langle 3 \rangle 3$. g^{-1} maps A_E onto B_O.

$\langle 4 \rangle 1$. We can apply g^{-1} to every element a of A_E. That's because $|Anc(a)|$ is even and therefore, a has at least a first ancestor. According to $\langle 2 \rangle 2$, this is only the case if $a \in g(B)$ and in this case we can apply g^{-1} to a.

⟨4⟩2. LET: $a \in A_E$. We have $|Anc(a)| = 2k$, with $k \in \mathbb{N}^*$ and $Anc(a)$ is of the form: $\langle x_0 = a, x_1 = g^{-1}(a), \ldots, x_{2k-1}\rangle$. By definition, $Anc(g^{-1}(a)) = \langle g^{-1}(a) = x_1, \ldots, x_{2k-1}\rangle$ and we have $|Anc(g^{-1}(a))| = 2k - 1$. Hence, $g^{-1}(a) \in B_O$. We conclude that the function g^{-1} maps A_E in B_O.

⟨4⟩3. LET: $b \in B_O$. Then $Anc(b) = \langle x_0 = b, \ldots, x_{2k}\rangle$, with $k \in \mathbb{N}$. PICK $a = g(b)$. By definition, $Anc(a) = \langle a, g^{-1}(a) = b = x_0, \ldots, x_{2k}\rangle$, $|Anc(a)| = 2k + 2$ and hence $a \in A_E$ with $g^{-1}(a) = b$. We conclude that the function g^{-1} maps A_E onto B_O.

⟨2⟩7. DEFINE:

$$\phi(x) = \begin{cases} f(x) & \text{if } x \in A_I \cup A_0 \\ g^{-1}(x) & \text{if } x \in A_E \end{cases}$$

Then, ϕ is a one-to-one function on A with range B.

That's because of Step ⟨2⟩6 and the fact that $A = A_E \uplus A_O \uplus A_I$ and $B = B_E \uplus B_O \uplus B_I$.

⟨2⟩8. A and B are equipollent.

By Step ⟨2⟩7 and the definition of equipollent sets.

⟨1⟩3. CASE: A and B are not disjoint.

⟨2⟩1. If A and B are not disjoint, LET: $B' = \{A\} \times B$. B' is disjoint from A.

⟨2⟩2. There is a one-to-one function ϕ_1 on A with range B'.

⟨3⟩1. There is a one-to-one function f' on A to a subset of B'. There is a one-to-one function g' on B' to a subset of A.

⟨4⟩1. DEFINE: $f' : A \to B'$ by $f'(a) \mapsto A \times f(a)$.

⟨4⟩2. DEFINE: $g' : B' \to A$, by $g'(A \times b) \mapsto g(b)$.

⟨3⟩2. Since A and B' are disjoint, with the mentioned functions f' and g' we can use the proof of Step ⟨1⟩2 to construct a one-to-one function ϕ_1 on A with range B'.

⟨2⟩3. There is a one-to-one function ϕ_2 on B' with range B.

DEFINE: $\phi_2 : B' \to B$ by $\phi_2(A \times b) \mapsto b$.

⟨2⟩4. DEFINE: $\phi = \phi_2 \circ \phi_1$. Then ϕ is a one-to-one function from A with range B.

⟨2⟩5. A and B are equipollents.

By Step ⟨2⟩4 and the definition of equipollent sets.