Improving the User Experience of Proof Assistants: A Comprehensive Study of Interface Design and Accessibility

The Mistrust of Formal Proofs in Pure Mathematics

A Thesis Prospectus
In STS 4500
Presented to
The Faculty of the
School of Engineering and Applied Science
University of Virginia
In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science in Computer Science

By
Jamie Fulford

November 8, 2024

Technical Team Members: N/A

On my honor as a University student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments.

ADVISORS

Caitlin D. Wylie, Department of Engineering and Society

Rosanne Vrugtman, Department of Computer Science

Introduction

Software failures in critical systems have resulted in catastrophic consequences. The Therac-25 radiation therapy machine fatally overdosed patients due to race conditions in its control software, while the Ariane 5 rocket's self-destruction, triggered by an integer overflow, resulted in a $370 million loss. As software systems increasingly control critical infrastructure–from medical devices to financial systems to autonomous vehicles–preventing such devastating failures has never been more crucial. Proof assistants, specialized software tools that help develop and verify formal mathematical proofs of program correctness, offer a potential solution to this challenge. As Harrison et al. explain in their comprehensive history of the field, these tools evolved from early automated theorem provers into interactive systems that combine human insight with machine-verified mathematical rigor (Harrison et al., 2014). A formal proof, in this context, is a complete mathematical demonstration of correctness that can be mechanically verified by a computer, providing a level of certainty far beyond traditional testing methods. These powerful tools face a fundamental challenge: while they could revolutionize how we verify both mathematical theorems and software systems, there exists a wide gap between their theoretical potential and practical adoption. This gap manifests in both technical barriers to entry and deep social resistance, creating what Kiiskinen describes as a "curiously empty intersection" between proof engineering and computational sciences (Kiiskinen, 2023).

The implications of these problems extend far beyond academia. My technical topic investigates the accessibility barriers in proof assistant interfaces that prevent widespread adoption, while my STS research examines the deep-seated mistrust of formal proofs within the mathematics community. As software systems become more complex and interconnected, the cost of software failures grows exponentially. Traditional testing methods, while valuable, cannot provide the absolute certainty that formal verification offers. Yet without addressing both the technical barriers to usage and the social barriers to trust, proof assistants will remain underutilized precisely when they are most needed. By investigating and addressing both of these challenges, these powerful tools can

become more accessible and accepted, ultimately contributing to more reliable software systems and mathematical proofs in an increasingly software-dependent world.

Technical Topic

The inaccessibility of proof assistants represents an equity problem in software verification. While these tools are essential for ensuring software correctness and safety, their current interfaces create insurmountable barriers for many potential users, effectively excluding them from participating in formal verification work. This systematic exclusion manifests through unnecessarily complex interfaces and poor user experiences, restricting these powerful verification tools to a small group of specialists at a time when software verification is increasingly crucial for public safety and infrastructure.

The root cause of this problem lies in the academic origins of these tools. Volker provides a detailed analysis of this issue, demonstrating through an incentives analysis how the academic development context has led to systemic neglect of user experience. As he explains: ”There is little external reward for building and maintaining such user interfaces... most developers are academics or students” (Völker, 2004, p. 140). This has created a self-perpetuating cycle where tools are designed by and for formal methods experts, with little consideration for broader accessibility or usability. However, one might counter Volker's emphasis on academic incentives by pointing to successful open-source developer tools that emerged from academic contexts. Yet, as Kiiskinen demonstrates (Kiiskinen, 2023), proof assistants face unique challenges that make their development trajectory fundamentally different from other academic software projects. This analysis shapes my approach to interface design by suggesting that we must address both technical and institutional barriers to improvement. Koutsoukou-Argyraki documents how even experienced mathematicians struggle with these tools' basic syntax and proof exploration (Koutsoukou-Argyraki, 2021). The substantial time investment required creates what Kiiskinen calls a ”knowledge barrier” (Kiiskinen, 2023), particularly excluding applied fields that could benefit most from formal verification but lack resources to overcome these technical barriers.

Attempts to address these issues, such as the JsCoq project, have shown promise by making proof assistants more accessible through web interfaces (Gallego Arias et al., 2017). While Gallego Arias et al.'s JsCoq project represents a significant step forward in accessibility, their approach contrasts with Volker's recommendations for proof assistant interfaces (Völker, 2004). In particular, JsCoq maintains traditional interface paradigms, while Volker argues for more fundamental redesigns. This tension between incremental improvement and radical redesign informs my research approach.

To address this problem comprehensively, my research will employ three interconnected methods: First, I will conduct a systematic analysis of current proof assistant interfaces, documenting specific usability challenges through structured user interviews and task completion metrics with both expert and novice users. This analysis will build upon Harrison et al.'s (2014) framework for interactive theorem proving to identify specific pain points in modern interface design. Second, I will develop prototype interfaces that implement established HCI principles such as progressive disclosure and contextual help systems, focusing specifically on common proof construction workflows. These prototypes will incorporate successful patterns from modern programming IDEs, such as intelligent code completion and interactive error feedback. Finally, I will evaluate these prototypes through controlled user studies measuring specific metrics: time to complete standard proof tasks, error rates in proof construction, and user confidence ratings. The goal is to demonstrate that proof assistants can maintain their mathematical rigor while becoming significantly more accessible to new users, transforming how formal verification is integrated into both software development and mathematical practice. This work addresses what Volker correctly identified as a systemic problem in proof assistant development, creating interfaces that serve not just experts but the broader community of potential users.

STS Topic

The widespread rejection of proof assistants by the mathematics community represents a fundamental tension between traditional mathematical practice and technological innovation in knowl-

edge production. While these tools offer new possibilities for mathematical verification, their adoption threatens established social and epistemological structures within mathematics. As Koutsoukou-Argyraki reveals through her detailed firsthand account as a mathematician attempting to adopt these tools, this rejection stems from a deeper conflict between how mathematicians construct and validate mathematical knowledge and how proof assistants require that knowledge to be expressed (Koutsoukou-Argyraki, 2021). This conflict has created a growing divide between computer-assisted and traditional mathematical practices, potentially fragmenting the mathematical community and hindering the advancement of both formal verification and pure mathematics.

The Social Construction of Technology (SCOT) framework reveals how different communities have constructed varying and often conflicting interpretations of these tools (Pinch and Bijker, 2012). Traditional mathematicians, who view proofs as vehicles for mathematical insight and understanding rather than just verification, often resist the rigid formalism of proof assistants. Bayer et al. capture this perspective in their analysis of generational differences in mathematical practice, showing how established mathematicians construct proof validity around concepts of insight and intuition (Bayer et al., 2022). It is important to note that while this source was submitted to Arxiv, it was also a notice from the American Mathematical Society and is authored by reputable mathematicians, so it is a reliable source. This construction stands in stark contrast to that of computer scientists and formal methods researchers, who view proof assistants as natural extensions of mathematical reasoning.

Kaliszyk and Urban document how large-scale formalization projects become sites of negotiation between different mathematical practices (Kaliszyk and Urban, 2016). Their analysis reveals how formalizing mathematics forces communities to confront fundamental questions about proof validity, concept representation, and authority—questions that reflect deeper social constructions of mathematical truth. Shulman provides additional insight into this tension through his analysis of emerging mathematical foundations, demonstrating how proof assistants are actively reshaping what kinds of mathematical questions can be asked and answered (Shulman, 2024). This reconstructive potential–the ability to open new mathematical territories–often goes unrec-

ognized by those who view proof assistants primarily as verification tools. The resulting disconnect between potential and perception further reinforces the social barriers to adoption. However, Koutsoukou-Argyraki's firsthand account (Koutsoukou-Argyraki, 2021) presents a markedly different perspective from Shulman's optimistic vision of proof assistants reshaping mathematics (Shulman, 2024). While Shulman sees these tools as potentially revolutionary for mathematical practice, Koutsoukou-Argyraki's experience suggests that without addressing fundamental usability and cultural barriers, such potential may remain unrealized. This contradiction highlights the gap between theoretical potential and practical reality that my research aims to address.

My research will examine these dynamics through several interconnected methodological approaches, guided by the Social Construction of Technology (SCOT) framework (Pinch and Bijker, 2012). Through this lens, I will analyze three primary sources of evidence. First, I will conduct a historical analysis of debates around proof automation and formalization, drawing on Harrison et al.'s (2014) comprehensive history to trace how different social groups have constructed their relationship with formal methods over time. Second, I will analyze case studies of contemporary formalization projects, focusing on moments of controversy that reveal underlying social constructions of mathematical practice. Third, building on Anand's framework for analyzing trust in proof assistants (Anand, 2016), I will examine how different communities construct notions of mathematical validity. Their analysis, while in their Ph.D. thesis, was successfully defended through peer review, so it is a reliable source. Moreover, the emergence of what Blanchette et al. term "hammer" systems–automated tools that attempt to bridge human and machine reasoning–further complicates these constructions (Blanchette et al., 2016). These investigations aim not simply to document resistance to proof assistants but to understand how different social groups actively construct and reconstruct their relationship with formal verification tools. By analyzing these various forms of evidence through the SCOT framework, we can better understand both the sources of resistance and potential paths toward greater acceptance and integration of formal methods in mathematical practice.

Conclusion

The intersection of poor user experience and social mistrust in proof assistants represents a significant barrier to realizing the full potential of formal verification in both mathematics and software development. Through my technical research into interface design and accessibility, I aim to demonstrate that proof assistants can become more user-friendly without sacrificing their mathematical rigor. My STS analysis, using the SCOT framework, will provide insights into how different social groups interpret and interact with these tools, helping bridge the gap between traditional mathematical practice and formal verification methods. Together, these investigations can contribute to making proof assistants more accessible and trusted, potentially transforming how we verify both mathematical theorems and critical software systems. As formal verification becomes increasingly important in our software-dependent world, addressing these technical and social barriers is crucial for ensuring the reliability and correctness of complex systems that impact everyday life.

References

Anand, A. (2016, August). *Trust in proof assistants: Opportunities and limitations* [Ph.D. Thesis]. Cornell University. https://doi.org/10.7298/X4BC3WGB

Bayer, J., Benzmuller, C., Buzzard, K., David, M., Lamport, L., Matiyasevich, Y. V., Paulson, L. C., Schleicher, D., Stock, B., & Zelmanov, E. I. (2022). Mathematical proof between generations. *ArXiv*, *abs/2207.04779*. https://api.semanticscholar.org/CorpusID:250426455

Blanchette, J. C., Kaliszyk, C., Paulson, L. C., & Urban, J. (2016). Hammering towards qed. *Journal of Formalized Reasoning*, *9*(1), 101–148. https://doi.org/10.6092/issn.1972-5787/4593

Gallego Arias, E. J., Pin, B., & Jouvelot, P. (2017). jsCoq: Towards hybrid theorem proving interfaces. In S. Autexier & P. Quaresma (Eds.), *Proceedings of the 12th workshop on user interfaces for theorem provers, coimbra, portugal, 2nd july 2016* (pp. 15–27, Vol. 239). Open Publishing Association. https://doi.org/10.4204/EPTCS.239.2

Harrison, J., Urban, J., & Wiedijk, F. (2014). History of interactive theorem proving. In J. H. Siekmann (Ed.), *Computational logic* (pp. 135–214, Vol. 9). North-Holland. https://doi.org/10.1016/B978-0-444-51624-4.50004-6

Kaliszyk, C., & Urban, J. (2016). Wikis and collaborative systems for large formal mathematics. In P. Molli, J. G. Breslin, & M.-E. Vidal (Eds.), *Semantic web collaborative spaces* (pp. 35–52). Springer International Publishing.

Kiiskinen, S. (2023). Curiously empty intersection of proof engineering and computational sciences. In P. Neittaanmäki & M.-L. Rantalainen (Eds.), *Impact of scientific computing on science and society* (pp. 45–73). Springer International Publishing. https://doi.org/10.1007/978-3-031-29082-4_3

Koutsoukou-Argyraki, A. (2021). Formalising mathematics – in praxis; a mathematician's first experiences with isabelle/hol and the why and how of getting started. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, *123*(1), 3–26. https://doi.org/10.1365/s13291-020-00221-1

Pinch, T., & Bijker, W. (2012). *The social construction of technological systems: New directions in the sociology and history of technology*. The MIT Press. Retrieved November 9, 2024, from http://www.jstor.org/stable/j.ctt5vjrsq

Shulman, M. (2024). Strange new universes: Proof assistants and synthetic foundations [Published electronically: February 15, 2024]. *Bulletin of the American Mathematical Society*, *61*, 257–270. https://doi.org/10.1090/bull/1830

Völker, N. (2004). Thoughts on requirements and design issues of user interfaces for proof assistants [Proceedings of the User Interfaces for Theorem Provers Workshop, UITP 2003]. *Electronic Notes in Theoretical Computer Science*, *103*, 139–159. https://doi.org/10.1016/j.entcs.2004.05.001