Communicating Formal Proofs: The Case of Flyspeck

Carst Tankink[1], Cezary Kaliszyk[2], Josef Urban[1], and Herman Geuvers[1,3]

[1] ICIS, Radboud Universiteit Nijmegen, Netherlands
[2] Institut für Informatik, Universität Innsbruck, Austria
[3] Technical University Eindhoven, Netherlands

Abstract. We introduce a platform for presenting and cross-linking formal and informal proof developments together. The platform supports writing natural language 'narratives' that include islands of formal text. The formal text contains hyperlinks and gives on-demand state information at every proof step. We argue that such a system significantly lowers the threshold for understanding formal development and facilitates collaboration on informal and formal parts of large developments. As an example, we show the Flyspeck formal development (in HOL Light) and the Flyspeck informal mathematical text as a narrative linked to the formal development. To make this possible, we use the Agora system, a MathWiki platform developed at Nijmegen which has so far mainly been used with the Coq theorem prover: we show that the system itself is generic and easily adapted to the HOL Light case.

1 Introduction

Formal proof development is gradually becoming accepted as a means for establishing the correctness of mathematical theory in particular. Large repositories of formal proof have been created in various proof assistants to prove impressive results, for example, the development of the odd order theorem in Coq [1], the proof of the 4 color theorem in Coq [2] and the proof of the Kepler conjecture [3] in HOL Light. A major issue is how to communicate these large formalizations: to people that want to cooperate or want to build further on the development, to people who want to understand the precise choices (of definitions and proofs) chosen in the formalization and to people who want to convince themselves that it is really the proper theorem that has been proven. At the moment, communicating a formal proof is hard, as can also be noticed from the fact that the number of publications about the impressive formalizations mentioned above is low. Moreover, these publications hardly give access to the formalization, but describe the project on a rather high level of abstraction. The Journal of Formalized Reasoning[4], the Archive of Formal Proofs[5] and the Journal of Formalized

[4] `http://jfr.unibo.it/`
[5] `http://afp.sf.net`

Mathematics[6] try to improve on this by explicitly giving a platform for formalizations (the latter for Mizar), but that is not really taking off.

In the present paper we present a wiki based approach towards the communication of large formal proof developments. Formal proofs are close to programs written in a high-level programming language, which need to be documented to be understandable and maintainable. However, a mathematical proof development is also special, because there (almost always) already is documentation, which is the mathematical document (a book or an article) describing the mathematics (definitions, notation, lemmas, proofs, examples).[7] This is what we call *informal mathematics* as opposed to *formal mathematics* which is the mathematics as it lives inside a proof assistant. These days, informal mathematics consists of LaTeX files and formal mathematics usually consist of a set of text files that are given as input to a proof assistant to be checked for correctness. Our approach is to provide tools that allow users to do the following.

1. One can automatically generate wiki files from formal proof developments. These wiki files can then be displayed in a browser, where we maintain all linking that is inherently available in the formal development (e.g. via definitions and applications of lemmas).
2. When hovering over the formal proofs, one sees the proof state at that point, so a reader can observe what the action of the proof commands is. This uses the Proviola technology that we have previously developed and described [4].
3. One can also automatically generate wiki files from a set of LaTeX files. These wiki files can then be displayed in a browser, where we maintain the linking inside the LaTeX files, but more importantly, also the linking with the formal proof development.
4. One can write a wiki document about mathematics and include snippets of formal proof text via an inclusion mechanism. This allows one to dynamically insert a piece of formal proof, by referencing the formal object in a repository, which is then automatically rendered and displayed inside the wiki document.

The tools we describe are part of the Agora system we are developing in Nijmegen, which aims at being a "Wiki for Formal Mathematics": a web platform to present and document formalizations, but also to cooperate on joint formalizations. With Agora, we want to lower the threshold for participating in formalization projects by:

– Providing an easy-to-use web interface to a proof assistant [5].
– Marking up formal mathematics for the Web without effort by authors [6], allowing users to browse this database for examples and inspiration.
– Providing tools for linking informal and formal text [7].

[6] `http://fm.mizar.org/`

[7] Formalizations of (software) systems typically have an informal *specification*, which serves a similar role as a mathematical document, and which could be served by the tools described in this paper, although some of the workflows described here might not match completely.

– Providing additional tools for users of proof assistants, like automation or proof advice.

The system is designed to support the dissemination of formal mathematics to an audience that does not necessarily have prior exposure to an interactive theorem prover. Our general claim is that this type of technology is crucial to further the field of formalized mathematics. One has to develop computer support for documenting and communicating formal proofs and for linking formal proofs to a high level 'narrative'.

Until recently, the system only fully supported formalizations written for the Coq theorem prover, having been tested on smaller test cases. However, the system is designed to be generic, reusing components that can be specialized for specific theorem provers. This paper describes the extension of Agora to include the HOL Light theorem prover [8], to allow the system to serve the files of the Flyspeck project in a wiki.

2 Presenting Flyspeck in Agora

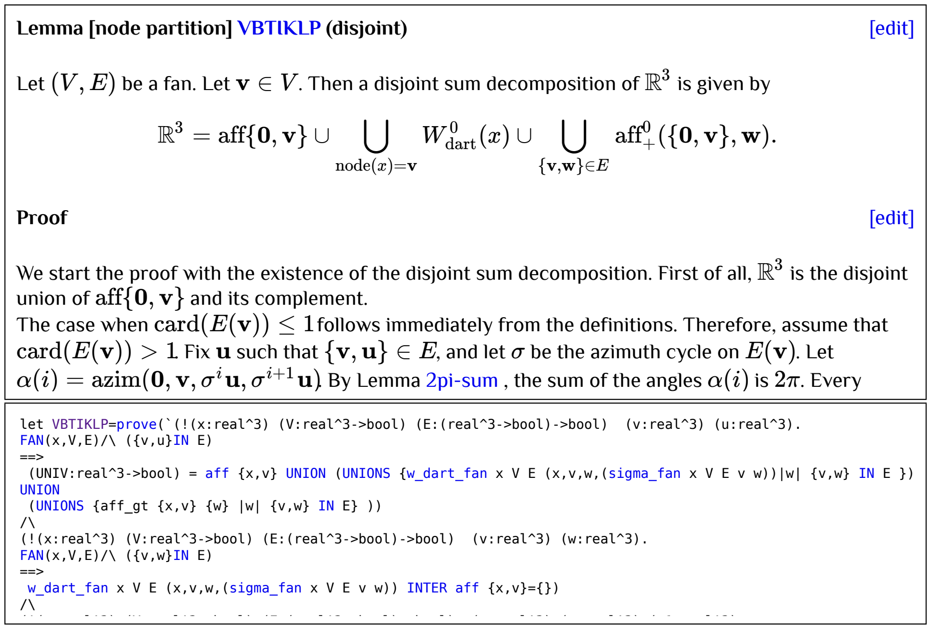
In the present paper we will not go into the general goals or design of Agora, but only show the tools that support the 4 activities mentioned above. We show the practical usability of the tools by presenting a page of the Flyspeck formal development in HOL Light, together with the page of the informal mathematical description (Figure 1). By discussing these pages, the links between them and how they have been created, we describe our tools.

An example document resides in Agora,[8] and is shown in part in Figure 1. For the best experience, we suggest the reader follows along at the demonstration page while reading this section. Implementing low-threshold interactive web-editing of formal HOL Light code is currently work in progress.

2.1 Description of a Formal Proof

The first noticeable feature of the document is that it is almost isomorphic to Chapter 5 of the text accompanying the Flyspeck formalization [9] (our source document). As mentioned above, important formalizations are not merely technical proof scripts: they go hand-in-hand with informal (in this case Hales's) mathematical narrative. To obtain the informal text, we have processed the LaTeX sources of Hales's text, transforming it into the Creole syntax [10]. This syntax is similar to Wikipedia's input language: a light-weight markup language that is easy to translate to HTML. The formulae in the source document are kept largely intact: they are processed at render-time by MathJax[9]: a JavaScript tool for rendering mathematics in a browser-independent way. This approach makes the resulting document editable as a wiki page written in Creole. A more complete approach would be to also accept LaTeX as input language for writing the documentation, something we intend to address as a follow-up.

[8] `http://mws.cs.ru.nl/agora_flyspeck/flyspeck/fly_demo`
[9] `http://mathjax.org`



Lemma [node partition] VBTIKLP (disjoint) [edit]

Let (V, E) be a fan. Let $\mathbf{v} \in V$. Then a disjoint sum decomposition of $\mathbb{R}^3$ is given by

$$\mathbb{R}^3 = \mathrm{aff}\{\mathbf{0}, \mathbf{v}\} \cup \bigcup_{\mathrm{node}(x)=\mathbf{v}} W^0_{\mathrm{dart}}(x) \cup \bigcup_{\{\mathbf{v},\mathbf{w}\}\in E} \mathrm{aff}^0_+(\{\mathbf{0}, \mathbf{v}\}, \mathbf{w}).$$

Proof [edit]

We start the proof with the existence of the disjoint sum decomposition. First of all, $\mathbb{R}^3$ is the disjoint union of $\mathrm{aff}\{\mathbf{0}, \mathbf{v}\}$ and its complement.
The case when $\mathrm{card}(E(\mathbf{v})) \leq 1$ follows immediately from the definitions. Therefore, assume that $\mathrm{card}(E(\mathbf{v})) > 1$. Fix $\mathbf{u}$ such that $\{\mathbf{v}, \mathbf{u}\} \in E$, and let σ be the azimuth cycle on $E(\mathbf{v})$. Let $\alpha(i) = \mathrm{azim}(\mathbf{0}, \mathbf{v}, \sigma^i\mathbf{u}, \sigma^{i+1}\mathbf{u})$. By Lemma 2pi-sum , the sum of the angles $\alpha(i)$ is 2π. Every

```
let VBTIKLP=prove(`(!(x:real^3) (V:real^3->bool) (E:(real^3->bool)->bool)  (v:real^3) (u:real^3).
FAN(x,V,E)/\ ({v,u}IN E)
==>
 (UNIV:real^3->bool) = aff {x,v} UNION (UNIONS {w_dart_fan x V E (x,v,w,(sigma_fan x V E v w))|w| {v,w} IN E })
UNION
 (UNIONS {aff_gt {x,v} {w} |w| {v,w} IN E} ))
/\
(!(x:real^3) (V:real^3->bool) (E:(real^3->bool)->bool)  (v:real^3) (w:real^3).
FAN(x,V,E)/\ ({v,w}IN E)
==>
 w_dart_fan x V E (x,v,w,(sigma_fan x V E v w)) INTER aff {x,v}={})
/\
```

Fig. 1. An informal proof together with its formal counterpart. Cropped screenshots from document pages at `http://mws.cs.ru.nl/agora_flyspeck/flyspeck/fly_demo`.

2.2 Integration with Formal Proof

A nicely marked up paper, whether or not it appears as a Web page, is not a description of a formal proof: for this, it needs to include parts of the formalization, in order to showcase and document them: this inclusion does not have to be complete, as this might muddle the description with details that are not immediately necessary for understanding. So, the second feature of the document is that the definitions and lemmas in it are surrounded by a box, and marked with buttons marked "formal" and "informal": using these buttons, a reader can toggle between the informal text of such a text and the corresponding formalization.

This functionality is made possible thanks to (Hales's) annotations of the source text, combined with a previously developed technology for Agora. For (almost) each island of 'mathematics' (definitions, lemmas, theorems...), the source text defines the corresponding entity or entities in the formal development. The corresponding entity can be included in the page using Agora's inclusion facility [7], by transforming the correspondence into a kind of hyperlink. The necessary syntax can be also hand-written in the wiki (this can be used for gradual addition of more and more cross-links to the formal code), but so far everything was generated from the source text annotations.

This approach differs from Isabelle/Isar [11], which supports a user in writing a formal proof and its documentation in a more literate way: the full proof is part of the document, and can be verified by the system. Agora's documentation tools, on the other hand, allow writing a documentation layer on top of the formal code within the system.

2.3 Dynamic Display

The Proviola tool integrated in Agora's rendering chain reduces the task of evaluating proof state to just pointing at parts of the proof: it shows a proof script as HTML[10] and when the user points at a particular command, the associated state is computed in the background, caching it for retrieval without computation at a later moment.. This interaction model has two advantages: (i) it eliminates the overhead of installing, configuring, and learning about a theorem prover; inspection of an interesting proof state or tactic is reduced to pointing a cursor at it, and (ii) it reduces the amount of task switches a reader will have to make.

Proviola has a generic design, and the inclusion of HOL Light for its batch task was simple: writing a parser that recognizes input to the prover, and some glue that allows the Proviola to send these commands to HOL Light, and read the output.

3 Conclusion and Future Work

Agora is an online platform that facilitates collaborative gradual formalization of mathematical texts, and allows their dual presentation as both informal and formal. In particular, the platform takes both LaTeX and formal input, cross-links both of them based on simple user-defined macros and on the formal syntax, and allows one to easily browse the formal counterparts of an informal text. One future direction is to allow even the non-mathematical parts of the wiki pages to be written directly with (extended) LaTeX, as it is done for example in PlanetMath. This could facilitate the presentation of the projects developed in the wiki as standalone LaTeX papers. On the other hand, it is straightforward to provide a simple script that translates the wiki syntax to LaTeX, analogously to the existing script that translates from LaTeX to wiki. We also still have to instantiate the interactive editing capability (now available for Coq) to HOL Light. This editing capability would allow a reader to directly edit the formal islands in the wiki pages by talking to a server-side HOL Light instance (with a reasonably advanced checkpointed state), loaded with additional prerequisities (in particular the previous formal islands). Allowing 'anyone' to edit does raise questions about how to maintain the integrity of a formalization. While we do not have an implemented solution to this in Agora, there are several options available [12,13] to deal with these issues, which still need to be evaluated on usability: an ideal solution needs to prevent a user thrashing the library, but does not shut out new users by overly complex or time-consuming protocols.

We are working on integrating the recently developed proof advice system [14] for HOL Light. The advisor uses machine learning to find lemmas that can be

[10] E.g., `http://mws.cs.ru.nl/agora_flyspeck/flyspeck/fan/fan_misc/index`

useful in solving a goal, encodes the goal together with the advised lemmas in TPTP format and runs a number of ATPs to find a minimal set of needed lemmas to then reconstruct the goal in HOL. This can be especially useful in a Wiki environment with many (possibly non-expert) contributing users, where it can be also used to automatically discover redundancies and refactor the formalization. Another direction is adding good linguistic techniques for translating informal texts to formal ones based on training on the annotated corpora (arising through this work). We could also try to include (or even better: track) informal wikis like ProofWiki, and start adding formal counterparts and annotations to them. Similarly for papers and books that were formalized in various systems: for example the books leading to the proof of Feit-Thomson theorem and their recent Coq formalization, and the Compendium of Continuous Lattices and its Mizar formalization.

References

1. Gonthier, G.: Engineering mathematics: the odd order theorem proof. In Giacobazzi, R., Cousot, R., eds.: POPL, ACM (2013) 1–2
2. Gonthier, G.: The four colour theorem: Engineering of a formal proof. In Kapur, D., ed.: ASCM. Volume 5081 of LNCS., Springer (2007) 333
3. Hales, T.C., Harrison, J., McLaughlin, S., Nipkow, T., Obua, S., Zumkeller, R.: A revision of the proof of the Kepler conjecture. Discrete & Computational Geometry **44**(1) (2010) 1–34
4. Tankink, C., Geuvers, H., McKinna, J., Wiedijk, F.: Proviola: A tool for proof reanimation. In: AISC/MKM/Calculemus. Volume 6167 of Lecture Notes in Computer Science., Springer (2010) 440–454
5. Tankink, C.: Proof in context — web editing with rich, modeless contextual feedback. To appear in proceedings of UITP 2012 (2012)
6. Tankink, C., McKinna, J.: Dynamic proof pages. In: ITP Workshop on Mathematical Wikis (MathWikis). Number 767 in CEUR Workshop Proceedings (2011)
7. Tankink, C., Lange, C., Urban, J.: Point-and-write. In: AISC/MKM/Calculemus. Volume 7362 of LNCS., Springer (2012) 169–185
8. Harrison, J.: HOL Light: An overview. In: Proceedings of the 22nd International Conference on Theorem Proving in Higher Order Logics, TPHOLs 2009. Volume 5674 of LNCS., Munich, Germany, Springer-Verlag (2009) 60–66
9. Hales, T.C.: Dense Sphere Packings - a blueprint for formal proofs. Cambridge University Press (2012)
10. Sauer, C., Smith, C., Benz, T.: Wikicreole: a common wiki markup. In: WikiSym '07. WikiSym '07, New York, NY, USA, ACM (2007) 131–142
11. Wenzel, M., Paulson, L.C.: Isabelle/isar. In: The Seventeen Provers of the World. Volume 3600 of Lecture Notes in Computer Science., Springer (2006) 41–49
12. Alama, J., Brink, K., Mamane, L., Urban, J.: Large formal wikis: Issues and solutions. In: Calculemus/MKM. Volume 6824 of LNCS., Springer (2011) 133–148
13. Corbineau, P., Kaliszyk, C.: Cooperative repositories for formal proofs. In: Calculemus/MKM. Volume 4573 of LNCS., Springer (2007) 221–234
14. Kaliszyk, C., Urban, J.: Learning-assisted automated reasoning with Flyspeck. CoRR **abs/1211.7012** (2012)