Alcor:
A user interface for Mizar

Paul Cairns

UCL Interaction Centre
University College London
London WC1E 7DP, UK
p.cairns@ucl.ac.uk

Abstract. The Alcor user interface to the Mizar library is intended to provide a test bed for exploring how a mathematician might interact with mathematical knowledge management tools. Specifically, how can a mathematician whilst working on or writing up mathematics look up relevant mathematical knowledge without interrupting their workflow? We describe how a specific interaction style has been used to implement keyword search and discuss how that style could benefit other more complex forms of context-specific searching.

1 Interfaces to MKM

The major goal of mathematical knowledge management (MKM) is to make available the enormous volume of mathematical knowledge available in current mathematical resources [9]. Of course, to make that knowledge valuable it needs to be provided to the right person, at the right time, *in the right way*. The potential user-base for MKM is almost anybody who applies or develops mathematical knowledge as part of the their work, that is, mathematicians, physicists, economists, chemists, social scientists and so on. It is therefore unlikely to be able to provide a single user-interface that would be acceptable to all users. The focus then in the current work is to look at user interfaces that might support mathematicians in developing mathematical material that could then integrate back in to MKM resources for other mathematicians. Even so, it is expected that mathematicians could produce and draw upon a wide-range of heterogeneous resources such as web-sites, journal articles, software and text-processing documents such as LaTeX.

This general task of integrating knowledge across distributed, heterogeneous resources with multiple authors is a typical knowledge management task[8]. However, it is further complicated in the case of mathematics that any synthesis of this material that a mathematician might make and build upon needs to be re-integrated back into the body of mathematical knowledge. There is no room for imprecision or personal interpretation of meaning as this could mean the difference between a correct proof and an incorrect one.

Mathematical knowledge management is still some way from achieving such an integrated environment for the working mathematician. I accordingly do not

try to address the full problem but instead consider a mathematician who might be working using some homogeneous library of mathematics and who would like in turn to contribute their work back to the library. The prototype environment would be an author for the Mizar Mathematical Library [6]. To be a successful Mizar author, a person must add an article to the library that constitutes some significant mathematical contribution to the library and in addition it should avoid replicating or apparently contradicting the definitions and theorems already in the library. To explore the working activities of such an author, the Alcor system has been developed as a possible working environment. This paper describes the main motivations for developing the Alcor system. The current system falls considerably short of these lofty goals, nonetheless, we describe its current features and capabilities before concluding with the planned future developments.

1.1 A note on the name

The Mizar system is a proof checking environment built on a sophisticated mathematical vernacular. It is famously named after the second star (in fact, quadruple star system) in the handle of the constellation *Ursa Major*, also designated ζ UMa. Interestingly, Mizar has a close companion star, Alcor, not in the main Mizar system. Alcor is visible to the naked eye for those with good acuity. Native Americans actually called the pair of stars the horse and rider [7]. Inasmuch as the system described here proposes to sit alongside the Mizar system rather than augment or integrate into it, the name Alcor seemed appropriate in many ways.

2 Using Mizar

Though the motivation in developing Alcor was to explore how a working mathematician might use Mizar or a system like Mizar, a user-centred design approach that would be typical for user interface development [11] has not been adopted. This is for the simple reason that there are currently very few working mathematicians who regularly use any sort of automated proving or proof checking system. The proposed developments in this system and in mathematical knowledge management more generally are innovative, fitting into a complex working pattern [8] and so a study of existing practices would, at this stage at least, be uninformative. Instead, a problem statement is used to encapsulate the goals of the system whilst not forgetting the priority of the user [10]:

> Design a graphical user interface to enable mathematicians in their ordinary work to successfully develop Mizar articles that do not replicate existing Mizar content and without the need for extensive knowledge of the entire Mizar library.

The approach to addressing this problem statement is motivated by a personal view of the difficulties in developing a Mizar article. A full discussion of

Mizar and the process of developing articles would be inappropriate here and additionally it has been discussed in great detail elsewhere [12,15]. In brief, though, a Mizar article is a collection of theorems, definitions and proofs written in the Mizar language. The language is very rich, particularly in comparison to other formal mathematical languages [14], being intended as a mathematical vernacular though this language is not without its complexities [4]. Having written theorems and proofs in the Mizar language, the Mizar system itself checks the proof and is able to assert whether or not the proof is formally correct within the definition of the language, the context of the particular article and any elements of the library referenced in the article. The goal, then, of any author is to produce theorems that are significant mathematically but not previously present in the library.

There are some natural barriers to achieving this goal. The library is very large with some 2000 definitions and 30,000 theorems. Full knowledge of it may only come with years of experience. In addition, like all formal mathematical languages there is a degree of verbosity [14] which means that not all theorems are of mathematical significance. Finding the key theorems for a particular task could be difficult. Also, like all formal languages, such as programming languages, it can sometimes be easier at first to copy and then adapt existing proofs. Even then, it is important to know that the terms in the proofs are being used according to their correct definitions.

Given this goal and these possible barriers, a person wishing to write a new article may well ask the following questions:

1. What is already in the library that I might need?
2. Can I adapt a proof of a similar statement?
3. Is there a theorem that I could use to prove this...?
4. What is the exact definition of...?

These first two questions work at a very high level being very much about the topic of work. The last two questions are more about the pragmatics of producing a correct Mizar proof. In particular, I am also working on the third question using LSI to provide a new search technique that may eventually address questions 1 and 2 [3]. The last question is relatively straightforward and it is this that is directly addressed in the current version of the Alcor system. It is hoped that as work on LSI progresses, Alcor will increasingly support answering the other three questions.

The need for better search facilities is recognised by frequent users (and developers) of Mizar [2], indeed, Bancerek and Rudnicki ask similar questions when it comes to searching Mizar. Their approach has been to develop a query language to do semantic retrieval that addresses the last two of the above questions. With the Alcor system, the approach has been rather to take a more graphical "point-and-click" style though at the moment this is at the cost of being less powerful in the kinds of retrieval that it can do.

3 The Alcor System

Given the current goal of the Alcor system is to be able to find definitions for an author working on an article, a workflow approach to authoring was taken inspired by the Phrasier system [5]. More explicitly, the author should have a full text editor that allows them to produce an article but at the same time be integrated with a search tool that enables them to look up definitions without interrupting their attention from their own work.

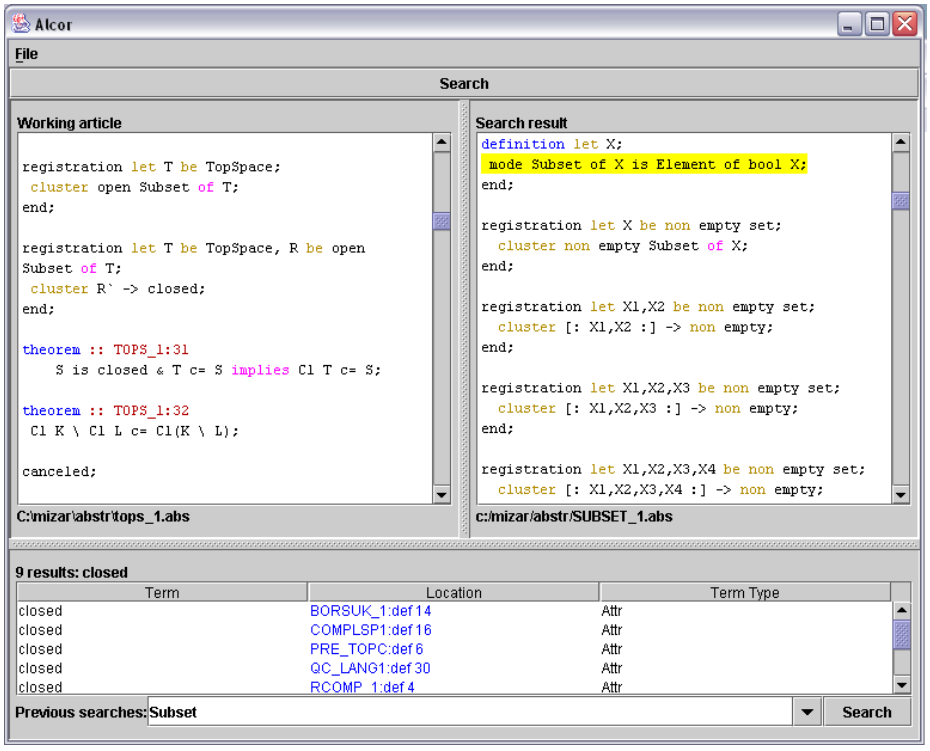

Fig. 1. The Alcor system

Figure 1 shows the main screen of the Alcor system. The left hand window is the main editor where the author would be working. There are many ways to enable a simple keyword search for definitions. Several have been included. There is a text box along the bottom of the screen where users can type free text. This text box also has a drop down menu that shows past search terms. This acts like a history function so that the author can easily replicate previously performed searches and peruse them if they have forgotten the exact terms of the search. The third method for entering search terms is simply to highlight words in the editor window. In this way, the author can maintain their focus at the position in

their article where they are currently working. Having highlighted the word, they can then click the "Search" button that runs beneath the menu bar (though this in fact is not easily recognised in its current location). Alternatively, in keeping with the PC paradigm, the user could right click and select to search on the highlighted word.

Search results appear in the lower window. Currently the display results are the location, that is, the Mizar reference, for the found items and also the type of item that has been found. This is because a single term may be defined in multiple senses depending on the particular need in particular articles. For example, the figure shows the results of a search on the term "closed" that has resulted in 9 items found. The context of work is an article on topology but of course with a novel piece of mathematics, it could be misleading to assume that only the topological definition of closed is required. For example, if the article were on topological groups then closed could be needed in either the sense for topologically closed or for algebraically closed.

To see the actual definitions, the user simply clicks on the location in the search results window. The required definition is displayed in the right hand window as given in the abstract (rather than the full Mizar article). This is partly simply to demonstrate the principle and partly because of implementation limitations in finding the definition in the full article. To make the definition clear to the user, it is positioned at the top of the window and highlighted in yellow.

As is often the case, the results of one search may lead on to further searches to clarify or explore other possibilities. The right hand window therefore also supports the same interactions for performing searches. Thus, the author could highlight search terms in their own work, get a retrieved abstract in the other window and carry on highlighting new search terms in the retrieved abstract. New retrieved abstracts would still come up in the right hand window so that the author would always have their own work on display.

Thus, a keyword search for definitions is implemented in a traditional graphical user interface style. In this sense, it is very like the Phrasier system where a text document being written, say a newspaper article, acts as a context for searching for similar or related articles. The found articles are shown in parallel to the article being worked on. In a natural language context, this could act as the "plagiarist's dream" (I. Witten, personal communication, 2002) but for mathematics where there is much less profit in plagiarism this workflow may be of real benefit.

Indeed, it is hoped that this has a more integrated workflow than the use of either a separate query language or a separate tool entirely such as `grep`. Also, the style of interaction is clearly extensible to other sorts of search. For example, to find a theorem that might help in proving a particular statement, the author could highlight the statement to be proved (rather than just a single word) and search on that. And highlighting a theorem might search for related theorems whose proofs could be adapted to prove the author's statement. In all of these examples, the context of the current work, namely the article that the author is currently working on, indicates the kind of query that is required. This removes

the onus from the author to formulate the query precisely and so allows them to concentrate only on authoring.

Of course, achieving this context-aware searching requires appropriate underlying algorithms. This is the aim of other work [3] but it is worth noting that the interaction style is actually independent of the methods use to produce good search results.

4 Future Work

In its current state, the Alcor system is intended as a test bed for different types of interaction that might support an author. In particular, the initial focus has been on supporting searching the Mizar Library. Thus, Alcor is far from a complete authoring environment for Mizar articles — it certainly does not attempt to offer the full functionality that the Mizar mode in emacs offers. In fact, so far, it only colours keywords in approximately the same way as the emacs Mizar mode. A first step before trying out Alcor with real Mizar authors would be to augment it to be at least as good as the emacs mode with the additional interactions discussed here. That way, authors would not be disadvantaged over using existing tools and and investigation of the real efficacy of the proposed interactions could be done.

The fact that searches produce retrieved articles that can then be used to define further searches means that there is the possibility of an author losing useful searches whilst they explore other possibilities. There is already a history mechanism for search terms but a more sophisticated history mechanism including forward and back as in a web browser and found items listed by article may also be required.

Alcor is intended to support *novel* search methods for the Mizar library and my other work is looking at how to implement that. A specific problem with the LSI method is how to formulate suitable queries so it is hoped that Alcor would provide sufficient support to automatically formulate queries based on the context of searching. In fact, the highlighting style of defining search terms could give valuable information as to the exact nature of the search being done. In this sense, the style of interaction would become integral to the search algorithm. This may not be desirable in the long run and other methods of defining context may be explored in future versions of Alcor.

References

1. Asperti, A., Buchberger, B., Davenport, J. H.: Mathematical Knowledge Management, Proceedings of MKM 2003. LNCS **2594** Springer Verlag (2003)
2. Bancerek, G., Rudnicki, P.: Information Retrieval in MML. In [1] (2003) 119–132
3. Cairns, P.: Informalising Formal Mathematics. To appear in MKM 2004, 3rd Int. Conf on Mathematical Knowledge Management (2004)
4. Cairns, P., Gow, J.: Using and Parsing the Mizar Language. Electronic Notes in Theoretical Computer Science **93** Elsevier (2004) 60–69

5. Jones, S., Stavely, M. S.: Phrasier: a system for interactive document retrieval using keyphrases. *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval* (1999) 160–167
6. Journal of Formalised Mathematics
`http://mizar.org/JFM`
7. Menzel, D.: A Field Guide to the Stars and Planets. Collins (1964)
8. Mirel, B.:Interaction Design for Complex Problem Solving. Morgan Kaufmann (2004)
9. MKM 04 web-site, accessed 9th August, 2004
`http://www.mizar.org/MKM2004/`
10. Newman, W. M., Lamming, M. G.: Interactive System Design. Addison-Wesley (1995)
11. Preece, J., Sharp, H. and Rogers, Y.: Interaction Design. Wiley (2002)
12. Rudnicki, P.: An overview of the Mizar project. In Proceedings of 1992 Workshop on Types and Proofs for Programs (1992)
13. Wiedijk F.: The DeBruijn Factor. Note accessed 9th August, 2004
`http://www.cs.kun.nl/ freek/factor/factor.pdf`
14. Wiedijk F.: Comparing Mathematical Provers. In [1] (2003) 188–202
15. Wiedijk F.: Mizar: An Impression. Note accessed 9th August, 2004
`http://www.cs.kun.nl/ freek/mizar/mizarintro.pdf`