System Description:
MBASE, an Open Mathematical Knowledge Base

Andreas Franke and Michael Kohlhase

FB Informatik, Universität des Saarlandes
afrank[~]kohlhase@ags.uni-sb.de

Abstract. In this paper we describe the MBASE system, a web-based, distributed mathematical knowledge base. This system is a mathematical service in MATHEMT that offers a universal repository of formalized mathematics where the formal representation allows semantics-based retrieval of distributed mathematical facts.

1 Introduction

Around 1994, an anonymous (but well-known) group of authors put forward the “QED Manifesto” [QED95], which advocates building up a mathematical knowledge base (and supporting software systems) as a kind of “human genome project” for the deduction community. Unfortunately, the vision has failed to catch on in spite of a wave of initial interest. In our view this is largely due to the lack of supporting software, as well as to the ensuing debate on the “right” logical formalism.

In this paper we describe the MBASE system, a web-based mathematical knowledge base (see http://www.mathweb.org/mbase). It offers a infrastructure for a universal, distributed repository of formalized mathematics. Since it is independent of a particular deduction system and particular logic\(^1\), the MBASE system can be seen as an attempt to revive the QED initiative from an infrastructure viewpoint. The system is realized as a mathematical service in the MATHEMT system [FK99], an agent-based implementation of a mathematical software bus for distributed theorem proving.

We will start with a description of the system from the implementation point of view in the next section (we have described the data model and logical issues in [KF00]). In section 3, we will take a brief look at the interface protocols based on the OPENMATH and KQML standards (see [FHJ+99,Koh00]). This reliance of Internet standards for communication makes MBASE an open system, and the implementation presented in this paper just one of its possible instances.

2 Architecture and Implementation

The MBASE system is realized as a distributed set of MBASE servers (see figure 1). Each MBASE server consists of a Relational Data Base Management

\(^1\) See [KF00] for the logical issues related to supporting multiple logical languages while keeping a consistent overall semantics.
System (RDBMS) e.g. Oracle connected to a Mozart process (yielding a MathWeb service) via a standard database interface (in our case JDBC). For browsing the Mbase content, any MBase server provides an http server (see http://mbase.mathweb.org:8000 for an example) that dynamically generates presentations based on HTML or XML forms.

This architecture combines the storage facilities of the RDBMS with the flexibility of the concurrent, logic-based programming language Oz [Smo95], of which Mozart is a distributed implementation (see http://www.mozart-oz.org). Most importantly for MBase, Mozart offers a mechanism called pickling, which allows for a limited form of persistence: Mozart objects can be efficiently transformed into a so-called pickled form, which is a binary representation of the (possibly cyclic) data structure. This can be stored in a byte-string and efficiently read by the Mozart application effectively restoring the object. This feature makes it possible to represent complex objects (e.g. logical formulae) as Oz data structures, manipulate them in the Mozart engine, but at the same time store them as strings in the RDBMS. Moreover, the availability of “Ozlets” (Mozart functors) gives MBase great flexibility, since the functionality of MBase can be enhanced at run-time by loading remote functors. For instance complex data base queries can be compiled by a specialized MBase client, sent (via the Internet) to the MBase server and applied to the local data e.g. for specialized searching (see [Duc98] for a related system and the origin of this idea).

MBase supports transparent distribution of data among several MBase servers (see [KF00] for details). In particular, an object $O$ residing on an MBase server $S$ can refer to (or depend on) an object $O'$ residing on a server $S'$; a query to $O$ that needs information about $O'$ will be delegated to a suitable query to the server $S'$. We distinguish two kinds of MBase servers depending on the data
they contain: archive servers contain data that is referred to by other MBases, and scratch-pad MBases that are not referred to. To facilitate caching protocols, MBASE forces archive servers to be conservative, i.e. only such changes to the data are allowed, that the induced change on the corresponding logical theory is a conservative extension. This requirement is not a grave restriction: in this model errors are corrected by creating new theories (with similar presentations) shadowing the erroneous ones. Note that this restriction does not apply to the non-logical data, such as presentation or description information, or to scratchpad MBases making them ideal repositories for private development of mathematical theories, which can be submitted and moved to archive MBases once they have stabilized.

3 Interface Language

The primary interface language of MBASE is the XML-based markup language ODoc [Koh00], a document-centered extension of the emerging OPENMATH standard [CC98] for mathematical objects. For instance the definition of a double function would be of the following form.

```xml
<definition id="double.def" item="double.sym" type="simple">
  <CMP xml:lang="eng">The doubling function defined by addition</CMP>
  <FMP><OMOBJ><OMBIND>
    <OMS cd="stlc" name="lambda"/>
    <OMVAR><OMV name="X"/></OMVAR>
    <OMA><OMS cd="arith1" name="plus"/><OMV name="X"/><OMV name="X"/></OMA>
  </OMBIND></OMOBJ></FMP>
</definition>
```

The CMP (commented mathematical property) element gives an informal characterization of the definition (which is a simple definition for the symbol with the identifier double.sym according to the attributes to the definition element) and the FMP (formal MP) gives the defining λ-term λX.(+X X) in OPENMATH representation. Note that the question of the semantics of such a term is determined by that of the symbols λ and +. These are specified in the MBASE theories given in the cd attributes of the OMS elements (the name of the symbol together with the theory establish unique reference in MBASE).

As a consequence of the XML-based approach it is possible to generate other logical formats from ODoc by specifying simple XSL [Dea99] style sheets; in fact the transformation from ODoc to the input formats of the OMEGA [BCF+97] and INKA [HS96] theorem provers is realized this way. It should be an easy exercise for most other concrete input formats. Furthermore one can generate customized ODoc documents from MBASE, which can then be presented in one of the more standard presentation media (e.g. \LaTeX or HTML/MathML).

Generating ODoc from a reasoning system is also quite simple in practice, since ODoc has a relatively simple structure (fully specified in an XML document type definition [Koh00]) that closely follows the term structure of OPENMATH.
(using the OMS, OMW, OMA, OMBIND elements to describe formula trees made up of symbols, variables, applications and abstractions).

4 Conclusion, Evaluation and Future Work

We have described the MBase system, a distributed mathematical knowledge base, it can be obtained from http://www.mathweb.org/mbase. This system differs from other repositories of mathematical data such as the Isabelle [Isa] or PVS [PVS] libraries in that it is an independent system not tied to a particular deduction system and offers inference services (matching, type-computation, ...). The data format is not geared towards a particular application.

It is currently used by the OMEGA and InKa theorem provers for storing and sharing logical theories including theorems, definitions, tactics and methods. In particular, the MBase service can be used as an ontology server fixing the semantics of mathematical objects used in protocols for deduction system integration. Furthermore, the MBase system is used as the basis of an interactive personalized mathematics book (IDA [CCS99]). Here, the structure information contained in the MBase version of the IDA data can be used to generate individualized sub-documents of IDA on the fly. While in the first case study the logical formulation of mathematical data is in the center of interest, in the second application textual representation plays a much more prominent role. MBase supports both formats and even fosters their integration.

The current implementation uses the very simple file-based gdbm database system. This is sufficient for the amount of data currently available in OMEGA, InKa and IDA. Furthermore it offers a very flexible, open and portable programming base. A version of MBase that uses ORACLE is currently under development.

Here a comparison to the MDB system [Har97] developed at the University of Erlangen is in order. MDB aims at supplying database support for the Mizar libraries, and is based on an object-oriented extension of ORACLE. Unfortunately, already the first 13 (of more than 300) articles already need 500 MB disc space in ORACLE. Our division of labor that treats logical formulae in the programming language MOZART and relational, text and structural data in a DBMS pays off here. The size of the database is only one order of magnitude larger than the size of the OMDoc encoding, which is comparable in size to the encodings used e.g. in OMEGA, Isabelle, or PVS. As an example for relative sizes of representations in MBase we consider the core theory library of OMEGA and the IDA text:

<table>
<thead>
<tr>
<th>Relative sizes of representations in MBase (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>System</td>
</tr>
<tr>
<td>OMEGA</td>
</tr>
<tr>
<td>IDA</td>
</tr>
</tbody>
</table>

Even when MBase implementations based on industrial strength relational data base systems like e.g. ORACLE are available, we believe that the current gdbm-based implementation can still serve as a local development knowledge base and
"proxy" system to ease the load on the central MBase repository servers. Such a local system will probably also be better suited to support the operations necessary for changing definitions and axiomatizations during the development of a theory.

In the current version, we have not yet treated more advanced structuring concepts like theory morphisms, inheritance wrt. signature mappings, etc. that have been developed for structuring the knowledge base (see [KF00]). There remains much to be done in this direction, and we hope to adopt techniques from algebraic specification (see for instance [Hut99]).

References


