MATHRESS: A MATHEMATICAL RESEARCH SYSTEM

Principal Investigator: Arnold Neumaier

Funding Period: 5 years

Part I: State-of-the-art and objectives

This project creates foundations for an automatic system that combines the reliability and speed of a computer with the ability to perform at the level of a good mathematics student. The acronym MATHRESS abbreviating the project title, which may be pronounced "mattress", indicates that the project serves to provide a good, comfortable foundation for the development of an automatic mathematical research system. The MATHRESS project creates the MATHRESS system that will itself be the foundation on which people will rely for mathematical support.

VISION and OBJECTIVES. The ambitious long-term vision for our project is the creation of an expert system that supports mathematicians and scientists dealing with mathematics in:
– checking their own work for correctness;
– improving the quality of their presentations;
– decreasing the time needed for routine work in the preparation of publications;
– quickly and reliably reminding them of work done by others;
– producing multiple language versions of their manuscripts;
– quickly disseminating partially checked results to other users of the system;
– intelligently searching a universal database of mathematical knowledge;
– learning like a student from the experience accumulated during interaction with the user.
Thus such a system will have a high value for every working scientist. In due course its functionality shall also be developed and extended such as to support and enable, by automatic real-time input and output conversion, visually impaired people to use the system and its mathematical content.

Our long-term vision therefore is to create an automatic general-purpose mathematical assistant MATHRESS that combines the indefatigability, accuracy and speed of a computer with the ability to perform at the level of a good mathematics student – both with respect to having the mathematical background knowledge and with respect to the ability to learn understanding ordinary mathematical language in a similar way as a student learns it; and also the ability to resolve standard exercises and give proofs. (Here understanding is meant in the sense that a competent user will find the system's answers consistent with the user's expectations.) Moreover, the MATHRESS should be efficient in supporting the mathematical modeling of real life applications involving large-scale scientific computing, by interfacing it to established problem solving environments at different levels of mathematical rigor, therefore making it attractive to industry mathematicians and scientists in various mathematically-supported fields.

It is planned to create the system in such a way that these benefits will begin to be available long before the full capability of the system is reached, thus making it attractive to mathematicians to use it and to contribute to its completion. This is done by creating a backbone that enables others to contribute already during the duration of the project. By developing the system under an appropriate open source license and by placing emphasis on a mathematician-friendly style of using the system, a web-based participation under a wiki-like construction will be attractive to students and professionals alike. To channel external work on the project, we consider ranking the parts of the database where help of contributors is needed most urgently.

The project will create a multilingual, multimedia, self-learning, easy-to-use interactive system that scientists will like to use because it provides mathematical contents and proof services on their laptops as easily as Google provides web services, Matlab provides numerical services, and Mathematica, Magma, GAP, or SAGE provides symbolic services, in a way that they can comprehend and that does not take undue amounts of extra time on their part. The system will allow users to work on a high, intuitive level; the need to interactively explain the intuition to the MATHRESS will have the well-known effect of enforcing clarity almost without effort.

The system is intended to be simple on the low level yet friendly on the high level. The former is achieved by building semantics into a very low-level representation, while the latter is accomplished by building high-quality interfaces. Combining low-level simplicity with high-level friendliness will be our guiding principle throughout the development of the project. MATHRESS will thus support not only mathematicians working in practice but scientists who use mathematics mainly as tools rather than being experts in developing these tools.

The present proposal is intended to achieve some essential milestones towards this vision, as developed below, not to fully achieve it within the 5 years of expected duration of the project.

A successful automatic mathematical research system requires:
– the creation of a database system for storing mathematical theory that contains a formal representation of the mathematical theory at the undergraduate level, and much more theory as the system grows;
– algorithms for reading and interpreting mathematical text specified in a LaTeX-like fashion (which is as close to natural mathematical language as possible);
– an analysis and efficient formalization of the communicative processes by which a human student acquires the capabilities to read and understand mathematical content;
– a flexible and easy to use human-machine interface;
– a dissemination and development plan for maximal impact on the scientific community;
– and much more, as detailed in Part II below.

A GRAND INTERDISCIPLINARY CHALLENGE. Our proposed work partly lies in the area of mathematical knowledge management, an accepted definition of which states: *"Mathematical Knowledge Management is an emerging interdisciplinary field of research in the intersection of mathematics, computer science, library science, and scientific publishing."* The project will thus be by definition highly interdisciplinary. It also draws on the field of Artificial Intelligence. The creation of a mathematical research system having cognitive capabilities of the kind envisaged in our project will require multidisciplinary research in a wide number of areas, some of them identified with clarity in Toby Walsh's never-realized 2003 British Grand Challenge in Computing proposal for a general-purpose mathematical assistant: **Databases:** a mathematical assistant will need to quickly access vast mathematical databases in complex ways (e.g., to search a database for a balanced incomplete block design with some given properties); **Knowledge representation:** a mathematical assistant will need a large ontology of mathematical information at both the object and the meta level; **Automated reasoning:** a mathematical assistant will need rich and complex inference mechanisms; **Learning:** a mathematical assistant will need to learn new mathematics; **User modeling:** a mathematical assistant will need to infer the user's goals and intentions from their actions; **Distributed computation:** a mathematical assistant will need to know how to break large computations down to tap into the Grid. We could add to this list several other areas of foremost and immediate relevance: **Computational Linguistics; Natural Language Processing; Foundations of Computing, Mathematics, Logic**.

Our project is also consistent with the aims and substance of the British "Cognitive Systems" Project launched in 2002 by the Foresight arm (`http://www.foresight.gov.uk/OurWork/CompletedProjects/Cognitive/index.asp`) of the UK Government, whose official definition (`http://mlg.eng.cam.ac.uk/cogsys/`) underlines well the interdisciplinarity of our research: *"Cognitive systems are natural or artificial information processing systems, including those responsible for perception, learning, reasoning, decision-making, communication and action. Cognitive Systems Engineering is a highly interdisciplinary field, drawing from disciplines as diverse as computer science, statistics, neuroscience, engineering, and psychology."*. With this definition in mind, the MATHRESS can be viewed as an artificial cognitive system for doing and understanding mathematics.

Having set above the scene for the proposal, we now give the breakdown on sections for the remaining of Part I:
— **"Existing Related Systems"**, in which the state-of-the-art is reviewed [pp. 3–6];
— **"MATHRESS in the Scientific Community"**, in which the defining features of our approach are contrasted with the state-of-the-art described in the prior section [p. 6];
— **"Specification of the Mathematical Research System"**, in which the high-level intended specification for our projected system MATHRESS is looked at in more detail [pp. 7–8];
— **"Expected Impact and Long-Term Consequences"**, in which the benefits and likely consequences of our project's envisioned success are discussed [p. 9–10].

EXISTING RELATED SYSTEMS. The PI's website (`www.mat.univie.ac.at/~neum/FMathL.html`) has a large selection of resources and references to existing related systems. For reasons of space we only refer to a selected handful below. There is a small-scale project that aims at formalizing elementary analysis rigorously at the level of a high-school student. The argument used to support the FEAR project (`http://www.nwo.nl/projecten.nsf/pages/2200126954`), whose results are due in March 2010, are in line with our own goal of automating the capability of an undergraduate student of mathematics: *"The main weakness of current proof assistants is that mathematically they are not powerful enough. They are very good at keeping track of all of the details of a mathematical proof: one routinely checks proofs using these systems that have hundreds, or even thousands, of cases. However, they are currently not very good at taking by themselves steps that a human mathematician considers to be trivial. One of the most important class of steps that are not supported well are problems that a high school student can solve without difficulty. We call these high school problems."* (`http://www.cs.ru.nl/~freek/notes/oc2004.pdf`)

Some available systems. There are already many automatic mathematical assistants that provide expert help in specialized domains. Known classes include computer algebra systems, automated deduction systems, modeling systems, matrix packages, numerical prototyping languages, decision trees for scientific computing software, etc.. Such existing tools already provide partial functionality of the kind to be created in the project but only tied to specific applications, or with a limited scope. The exemplifications below are extracted from the PI's paper "A modeling system for mathematics" (`http://www.mat.univie.ac.at/~neum/FMathL/project.pdf`), written for the ongoing MoSMath project, which comments on a large but not exhaustive list of such software systems, illustrating important features and limitations of what is currently available. **None can remotely approach the vision that we propose in this project.**

LaTeX (`http://www.latex-project.org`) is today's de facto standard for the creation, communication and publication of scientific documents, widely used by mathematicians, scientists, and engineers. It is fairly user-friendly and produces documents of excellent quality; however, it only specifies the syntax and the quotation structure (references to equations, theorems, tables, figures, papers, footnotes, endnotes, etc.) of the documents, leaving the semantics inaccessible.

Markup languages. OpenMath (`http://www.openmath.org`) is an extensible language for representing the semantics of mathematical objects as a structured text. Its purpose is to facilitate the exchange of mathematical information between computer programs, databases, or worldwide web documents, and to enable its display in a browser. But there is no intrinsic display form for OpenMath objects. MathML (`http://www.w3.org/Math`) is a low-level specification for describing mathematics as a basis for machine to machine communication, with emphasis on web applications. MathML deals principally with the presentation of mathematical objects (so that MathML generated display looks like nice ordinary mathematics). It only has limited facilities for dealing with content; but it can embed OpenMath constructs. OMDoc (`http://www.omdoc.org/omdoc`) is a semantics-oriented representation format and ontology language for mathematical knowledge extending the markup of OpenMath and MathML to the document and theory level of mathematical documents. The voluminous representations

```
<OMOBJ>
  <OMA>
    <OMS cd="calculus1" name="int"/>
    <OMBIND>
      <OMS cd="fns1" name="lambda"/>
      <OMBVAR> <OMV name="x"/> </OMBVAR>
      <OMA>
        <OMS name="sin" cd="transc1"/>
        <OMV name="x"/>
      </OMA>
    </OMBIND>
  </OMA>
</OMOBJ>
```

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
<matrix>
  <matrixrow>
    <cn> 0 </cn> <cn> 1 </cn> <cn> 0 </cn>
  </matrixrow>
  <matrixrow>
    <cn> 0 </cn> <cn> 0 </cn> <cn> 1 </cn>
  </matrixrow>
  <matrixrow>
    <cn> 1 </cn> <cn> 0 </cn> <cn> 0 </cn>
  </matrixrow>
</matrix>
</math>
```

of $\int \sin x \, dx$ in OpenMath and of the matrix $\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$ in MathML, taken from the above web sites, show that a markup format is not suitable as a modeling tool.

Mizar (`http://mizar.org`) is a formal language for specifying the syntax and semantics for mathematical texts in a way that allows them to be verified automatically by computer starting from set-theoretical axioms. A 16-year old journal entitled "Formalized Mathematics" specializes in publishing Mizar-generated mathematics. The Mizar input is difficult to read and create, but the output is reasonably readable, embedding the formulas in English (though often somewhat uneven) sentences. There are no capabilities to specify data-driven applications.

Common mathematical language. Making ordinary mathematical text completely comprehensible for the computer is an exceedingly difficult task, at present virtually impossible without much interactive assistence. Fully automatic partial work is reported in two Ph.D. theses by SIMON [14] and ZINN [18], which exhibit the many problems encountered in their attempts to turn some elementary number theory texts into checkable proofs. Therefore, the informal mathematical language must be somewhat restricted to be automatically analyzable. KAMAREDDINE [7] emphasizes the problems to be overcome for a full formalization acceptable to the typical mathematician, and introduces MathLang (KAMAREDDINE et al. [9, 10], VAN TILBURG [15]), a non-fully-formalized language close to the common mathematical language, for interactively annotating mathematics written in English and translating it into Mizar [8]. Currently, the system seems to be in an explorative stage of development and does not seem to have a structure that would lend itself easily to the specification of large optimization models or other applications.

Naproche (`http://www.math.uni-bonn.de/people/naproche`) is an ongoing project that aims to translate natural mathematical text into a sequence of first order logic sentences. The input for the system is a plain LaTeX file, without any further annotation. However, the system only accepts a very small and rigid controlled natural language. The people from the project are working on making this language more flexible, but until now it is by far not flexible and rich enough to be of actual use for a mathematician. The system produces also output in the FOF-format of the TPTP (`http://www.cs.miami.edu/~tptp/`), the standard challenge database for current theorem provers; hence, by using the tools of the TPTP, a wide variety of theorem provers can be accessed.

The Z notation. The Z notation (`http://vl.zuser.org`) is a specification language based on Zermelo-Fraenkel set theory and first order predicate logic, available since 2002 as an ISO standard [5]. It is a mathematical specification notation, used by industry as part of the software and hardware development process for the highly reliable definition of algorithmic components, particularly for safety-critical systems. Z is intended to increase human understandability of the specified system and to allow the possibility of formal reasoning and development. Professional quality typeset documents based on Z specifications can be produced by a LaTeX interface provided by the tool set CADiZ (`http://www-users.cs.york.ac.uk/~ian/cadiz/`); this system also provides support for finding proofs related to Z specifications. Various other tools are available, including an interface to the theorem proving environment HOL. However, there is no support for nonalgorithmic mathematics such as abstract infinite sets or integrals. Moreover, the specifications themselves look more like programs than like mathematics.

Modeling systems. For the application to numerical problems, in particular large-scale optimization, a number of versatile modeling languages are available; see KALLRATH [6]. Widely used examples include AMPL (`http://www.ampl.com`) and GAMS (`http://www.gams.com`). They are very efficient for problems from the applications, with interfaces to all major state of the art optimization packages, providing the derivatives needed by these solvers automatically. But the current modeling languages also have serious limitations; in particular, they are not able to understand arbitrary mathematical formulae.

CVX is a software system for disciplined convex programming, allowing the user to specify nonlinear optimization problems with automatic verification of their convexity properties, in an easy to use syntax. While very specialized, it is an example of a system that checks the assumptions of a numerical method before trying it, and by its syntax enforces a healthy disciplined approach to model formulation.

Decision trees for software. The Decision Tree for Optimization Software by MITTELMANN (`http://plato.la.asu.edu/guide.html`) gives online advice about which software to use for which kind of optimization problem. Similar decision trees are discussed in the literature for a variety of problems in scientific computing; see, e.g., ADDISON et al. [1], RAMAKRISHNAN & RIBBENS [12], RAMAKRISHNAN et al. [13], HOUSTIS et al. [4], MAARUSTER et al. [11], BUNUS [2]. Although not a modeling tool, such decision trees have a natural place in a modeling system that automatically selects the solver to use.

Monet (http://monet.nag.co.uk/monet) was a project undertaken by an international consortium of academic institutions and industrial partners between 2002–2004. Its objective was to develop a framework for the description and provision of web-based mathematical services. Monet differentiated between *static descriptions* of mathematical objects based on XML-based technologies such as OpenMath and MathML and *dynamic processing* of mathematical objects based on Description Languages such as WSDL. Monet produced interesting work, which can potentially be reused in our system, but its emphasis on intricate web technologies neither widely known nor very popular might have squandered its chances of practical success. In contrast, MATHRESS shall be a technologically minimalist project, based on what has been proven to work well in practice, which we believe is the "semantic wiki"-based approach illustrated e.g. by SWiM (http://www.kwarc.info/projects/swim).

Grammatical Framework. The Grammatical Framework (http://www.cs.chalmers.se/~aarne/GF) is a special-purpose programming system whose core is a generic grammar processor capable of recognizing and generating important fragments of many languages. It particularly addresses multilinguality, semantic conditions of well-formedness, language modularity and the reuse of grammars in different formats and as software components. It provides enough functionality for the restricted natural language support required for a formal language close to the traditional informal mathematical usage. Other language-related references of potential interest for the project are CARLSON et al. [3] for multilingual mathematics and Greenstone (http://www.greenstone.org) for multilingual software for building and distributing digital library collections.

WebALT (http://webalt.math.helsinki.fi) was a project funded by the eContent EU Programme between 2005–2007. It built upon the Grammatical Framework above in order to produce language-independent mathematical didactical material, targetting primarily students of undergraduate mathematics. It produced as its core deliverable a mathematical grammar for the language-independent encoding of exercises in mathematics, as well as mathematical grammars for a number of European languages to enable the automatic natural language generation of mathematical texts into these languages.

Wolfram|Alpha Computational Knowledge Engine (http://www.wolframalpha.com/) has recently (May 2009) launched to great expectation. It defines itself as a system for knowledge-based computing aiming to make "all systematic knowledge immediately computable by anyone". It is built on top of Wolfram Research's Mathematica system, although at present it does not accept general Mathematica input. Instead it accepts queries in natural language with formulas. The Wolfram|Alpha Computational Engine summons impressive computational capabilities; by contrast, its capability to understand and communicate with the user appears rather impoverished. Reasonable input in natural language is often not understood. Furthermore, the system is currently rather sensitive to changes in the syntax of the queries, which seems to be indicative of a lack of any real semantic comprehension.

Computer algebra systems and numerical prototyping languages. Mathematical software packages such as e.g. Mathematica (http://www.wolfram.com/products/mathematica), MuPAD (http://www.mupad.de) and Maple (http://www.maplesoft.com) are computer algebra systems with lots of built-in functionality. The systems Matlab (http://www.mathworks.com/products/matlab), Scilab (http://www.scilab.org) are programming environments for efficient numerical computation and prototyping, easy to use for the working mathematician. All these are widely used but do not give the user access to the mathematical meaning of the constructs, so that one cannot use the defined relations for anything but for computations.

Logical frameworks. There are a large number of logical frameworks and theorem proving systems which allow the verification of (carefully specified) proofs for mathematical statements, primarily of properties of programming languages and logics. Widely used systems include Coq (http://pauillac.inria.fr/coq), HOL light (http://www.cl.cam.ac.uk/~jrh13/hol-light), PVS (http://pvs.cl.sri.com), and Twelf (http://www.cs.cmu.edu/~twelf). Many of these systems can be used to prove nontrivial mathematics; see, e.g., WIEDIJK [16] (100 formally proved theorems, ranging from the fundamental theorem of algebra to Brouwer's fixed point theorem), and WIEDIJK [17] (a comparison of provers for the irrationality of $\sqrt{2}$). But proofs are generally unreadable manually (except by expert users). Typically, statements and proofs must be provided in a program-like structure far removed from mathematical practice. This makes these systems unsuitable for mathematical modeling applications.

Integrated 'Computer Algebra' and 'Automated Deduction' Systems. Beyond the specialized mathematical

assistants reviewed above, the issue of functionality integration is key to the creation of a system that has genuine general-purpose capabilities. This has traditionally been approached from two main directions, one connected with computer algebra systems and formal computation, the other concerned with automated deduction systems and theorem proving. The CALCULEMUS conference series has as its official objective the integration of these two classes of systems as a springboard for a more unified approach to the management of mathematical knowledge.

For instance, the **Theorema** package (`http://www.risc.uni-linz.ac.at/research/theorema`) is a Mathematica-based prototype extended by facilities for supporting logically rigorous proving of mathematical statements. It aims to assist the working mathematician through all the phases of the mathematical problem solving cycle (specification, exploration, proving, programming, and writing up). **Theorema** allows users to interactively build proofs for mathematical theorems, with nested-cell proofs that are readable and can be expanded or shortened in a user-controlled manner. Although input and output appear in a form acceptable for mathematicians, the results are often somewhat stilted. More importantly, there are no capabilities for specifying data-driven applications.

MATHRESS IN THE SCIENTIFIC COMMUNITY. In contrast to computer algebra or automated deduction systems reviewed above, our MATHRESS project grows organically from the breadth and depth of the PI's prior mathematical work on large-scale optimization and modeling packages leading to different insights. Our project benefits from the PI's prior experience as a working mathematician. Having worked with mathematics of many flavors, pure and applied, conceptual and algorithmic, foundational and data-driven, we bring a fresh perspective to a theme mostly dealt with by logicians or algebraists so far. **We open up a third way**; the intention is neither to compete with nor to criticize previous systems or existing approaches. In fact prior work from these sources is invaluable to the success of our project. MATHRESS will complement existing approaches to mathematical knowledge management and the formalization of mathematics, aiming to integrate them as we now explain.

The MATHRESS approach comes from a *different* side than the traditional logical and algebraic approaches. **The MATHRESS system aims at excellence in respects where all current systems are very poor, while remaining incomplete by itself where current systems are strong.** MATHRESS needs to be combined with suitable interfaces to reap the fruits of both. Our project complements what others already do independently, and lays the basis for a future self-supporting system in a Linux/Wikipedia fashion.

Since we need such a system for our applications to mathematical modeling, the sooner this is achieved the better. Thus we try to be complementary to what others do, working on the same goal from a different perspective – that of a pure and applied mathematician, and ensuring that something happens that really takes the working mathematician's perspective into account. Thus we believe that a successful automatic mathematical research assistant system needs both our efforts and those of the traditional partial and specialized approaches to the subject; therefore cooperation is the natural thing to aim at.

This project is going to build and cross the bridge between mathematicians, computer scientists, and logicians. There are a number of things which help distinguish the mathematics' perspective from the logic's perspective – mathematics is part of a social process, and this must be accounted for and modeled in the support system. For a mathematician, a proof is something quite different than for a logician - it should provide insight into *why* something is true, not just prove it correct. Pure correctness is an interesting but sterile fact. As Freek Wiedijk says in `http://www.cs.ru.nl/~freek/pubs/qed2.pdf`, the results of 140 man-years on the QED project (a well-known 1995 vision of the theorem proving community) in the traditional form would be a tar.gz file without any immediate further use; nice, but with not much gained.

Thus we will develop the tools that pave the way for logic proof systems to become useful for the ordinary mathematician who does not bother about constructivity, types, or the precise set theory, but just wants some assistance (i.e., relief, not an extra burden) for his daily work. This is what is severely lacking in all previous approaches. In particular, our plans involve an intention to work on the *partial* formalization of undergraduate mathematics. Thus current standard mathematics with their standard proofs (close to their textbook form, but made formally more precise) will be available as a codified set of theorems and formal proof sketches in a high-level formal language that we create. We organize mathematical content in computer-supported and community-supported ways, and integrate existing proof systems as they are without focusing on their development.

SPECIFICATION of the MATHEMATICAL RESEARCH SYSTEM. Our primary goal is to create a system that is useful to all scientists, and in particular, also to us. The creation of the system needs our expertise and knowledge as mathematicians. Despite their valuable work that we intend fully to exploit, we believe the proof assistant community has failed for many years to reach the mathematicians, the target community of first choice, because their priorities have been wrongly placed all along. The common mathematical language is not solely concerned with truth, but also with intelligibility, suggestiveness, understanding, simplicity, brevity, elegance, beauty, in a balance that finds the approval of the whole mathematical community. Scientists will want to use *this* language on the formal level, and reject having to learn an artificial language with additional notation without extra advantage. Logicians and computer scientists working on formalizing proofs have for long worked almost completely divorced from mathematicians. This is why one needs a group of mathematicians, not logicians or computer scientists, to change the situation. Having published in many different mathematical communities, the PI knows how mathematicians in quite diverse fields work and what they appreciate, and having also done much computer implementation work himself he knows how to translate this knowledge into software.

MATHRESS is intended to be:
— aimed at understanding, and not just representation, verification and retrieval (which constrain existing projects);
— learning from experience;
— extracting methods from examples;
— self-growing vocabulary, concept database, proving power;
— asking questions to improve or complete understanding;
— able to interface to automatic lemma discovery systems;
— coping with ambiguity (GLR parser and post analysis);
— coping with context dependence.
This intelligent flexibility will be necessary for a system aiding the formalization of mathematics. Conversely, some level of formalization of mathematics is indispensible for creating an automatic system with a useful level of understanding capability, and a clear and efficient communication and representation. Both aspects need therefore to be developed and deepened jointly and concurrently.

Data Acquisition in the MATHRESS. According to Barendregt & Wiedijk (`http://www.cs.ru.nl/F.Wiedijk/pubs/rspaper.pdf`), creating a complex mathematical document from the decision to write it (but after the results were already more or less obtained) to final publication currently takes about 4 hours/page. (In the PI's experience, this includes time needed for material selection, writing, correction, proofreading, reading and answering referee's reports, etc.) Creating a corresponding formally verified document with current systems takes, according to the same source, about 10 times as long, and the result is nearly indigestible for a human.

In contrast, MATHRESS will treat common mathematical language (or a slightly formalized version of it) as a declarative programming language for creating and modifying the contents of a reasoning system, reducing the role of the human supervisor to answering queries to disambiguate or explain – with time, fewer and fewer – statements that cannot be handled automatically, thus minimizing the human workload. Unlike in conventional systems, the language the automatic system understands will grow with every user interaction.

Our focus will be to stay close to actual practice in mathematics, working directly with their language, their editing tradition, and their informality; representing at first just what is already spelled out in textbooks, rather than building up water-tight logical links between all statements. The latter can be done in a separate, later step. We believe that **this feature distinguishes the proposed project from other people's plans**. MATHRESS allows users and external documents (textbooks, PlanetMath) to behave as trusted proof engines (which they often are – one can ask an expert to interpret a difficult passage in a textbook). Thus it has immediate value for documenting mathematics at the existing informal level. MATHRESS allows a gradual transition from informal to formal mathematics by replacing/adding formal proofs to informal signatures. We believe this is an important goal. We learn best when we have experience with concrete systems first. Those concrete systems can be handled informally to build up the intuition. Our approach is non-reductionist: instead of building everything up from a small low-level foundation that is more or less arbitrary (e.g. ZFC), we instead teach a machine the things that we do, on the level that we do it. Since we can, in most or at least in many cases tell quite exactly what we are doing, we gain confidence that we can also teach a machine to do the same.

Formalizing the Communicative Aspect of Mathematics in the MATHRESS. An automatic mathematical research system must have similar communication capabilities as a human mathematics student, with respect to both fellow humans and machines. The communicative processes by which students acquire and perfect such skills must therefore be modeled in detail and implemented. They form the backbone of the interaction abilities of the MATHRESS with one or more (human or automatic) agents. We will develop a formal model and a corresponding computer implementation of the social aspects of doing and communicating mathematics. In particular, the system must interpret text by itself: unlike current systems, we will not provide annotations and specifications for inaccurate text but give help only in the form of subroutines that generate automatically such annotations and specifications from the text as it is. Mathematics notation and proof style have been optimized over the centuries for an optimal compromise between information transfer and readability (given the required background) with the least amount of distraction. This feature will be preserved by our system. The best form of presentation is the one that minimizes work for the *human* reader. The human is the slowest, hence needs to be supported best. For example, `http://www.mat.univie.ac.at/FMathL/proofex.html` gives an instructive case of how our view of presenting a proof differs from that of the theorem proving community.

Formalizing Levels of Trust in the MATHRESS. To guarantee the highest possible level of reliability without forcing an unacceptable amount of rigor (and corresponding work) on every user, all mathematical units of content of our system will be assigned signatures containing information about what is assumed to be able to trust the statement. (A signature may say, e.g., "from Bourbaki's Elements", "imported from PlanetMath", "verified by the proof assistant HOL light on the basis of ZFC", "by an approximate Matlab calculation", "based on the Riemann hypothesis", or "unverified claim by user xyz".) There will be a trust propagation and verification system that recognizes which assumptions a given assertion is based on and, given a user-provided definition of what is trusted, points out all the untrusted links in existing verifications together with their signatures. This enables users to specify or even to experiment with the amount of trust they are prepared to put into various sources of information and deduction systems. Users interested in increasing the quality of the system can discover theories with important gaps on certain trust levels, and work towards closing those gaps. This metadata is designed to be as checkable as possible within the system.

Novel and Unconventional Aspects of the MATHRESS. As described above, our approach complements many existing tools, systems and interfaces or integrates them. Many individual pieces of the whole project have been considered often enough in the literature. This shows that the project is feasible – but no-one before or yet has managed or even tried to bring all issues together. Thus integration is one big unresolved issue that we address. The required integration needs a project focused in a *single* group rather than a web of independent contributors, however with close cooperation from the groups whose software we integrate. At the same time, MATHRESS will go far beyond the mere extension and application of known approaches in supporting the following **novel** aspects:

1. the systematic consideration of trust;
2. the modeling of the communicational aspects involved in doing and understanding mathematics;
3. the renunciation of fully formalized proofs as the primary requirement for a mathematical research system in favor of a maximally usable database of information immediately relevant to the working mathematician;
4. the formalization of the *claims* (theorems, corollaries, lemmata, statements appearing within proofs) of a paper in full detail, not primarily of their truth;
5. the faithful representation of *arbitrary* mathematics. By contrast, current systems fail to represent many much-used mathematical features (see `www.mat.univie.ac.at/~neum/FMathL.html#MathML`).

MATHRESS will likewise support a number of **unconventional** aspects:

1. the combination of rigorous techniques and heuristics to interpret ambiguous statements and uncover hidden contextual information;
2. storing a structural rather than a textual representation;
3. the project is intended as an integration framework for arbitrary theorem provers and solvers but does not do any theorem proving or solving itself;
4. emphasis is on beauty and ease of use rather than on formal verification;
5. in standard markup languages for mathematics, context information is unnecessarily repeated and makes the representation very voluminous (see p. 3 above). The intended representation in MATHRESS is almost the opposite, and has no more redundancy than a typical human mathematician would like to see.

EXPECTED IMPACT and LONG-TERM CONSEQUENCES. If our vision succeeds, the MATHRESS system will change the way mathematics is done and used in practice. The project shall create a system that people will like, and that makes the efforts accumulate easily. Then it grows by itself. Those more mathematically minded, concerned about using what is in the system for traditional, informal mathematics will develop that part, without bothering about more formality than necessary for them. Those more logically minded, concerned about skipped details will want to fill the gaps, add the missing conditions or degenerate cases, etc., to make the proofs fully rigorous, and find plenty of material prepared by the informal advance guard who scouted the territory. In this way, the approach taken in this project will suit both parties – the rigorists and the informalists, and, with time, will lead to convergence. The database of informal, partly formalized and fully formalized mathematics will be as useful to the computer-assisted scientist as ordinary mathematics libraries are for the ordinary scientist, and will soon become indispensible for the working mathematicians.

MATHRESS will enable the Large-Scale Formalization of Mathematics. We take a large step towards pushing the formalization of mathematics to the tipping point where it becomes large-scale, self-growing and mainstream.

Mathematicians know that they already have a very expressive language optimized for expressing things in any desired degree of brevity or detail, for maximal intelligibility and easy overview. The common mathematical language must be learnt by everyone anyway, hence it should be the basic language for any intelligent system designed for the ordinary scientist. Furthermore, the intrinsic error-detecting and error-correcting capabilities of common mathematical language are important, e.g., in mathematics digitization projects that involve the use of Optical Character Recognition (OCR). It is well-known that OCR errors may produce incomplete or slightly wrong text, which requires subsequent manual intervention to correct. It has been estimated (`http://eric.ed.gov/ERICDocs/data/ericdocs2sql/content_storage_01/0000019b/80/15/1c/60.pdf`) that the costs involved in creating a digital text file suitable for searching in the JSTOR database are even greater than the costs involved in the primary scanning step because of the necessity of corrective human intervention post OCR. In turn, mathematics digitization projects are substantially based on the semiautomatic production of LaTeX code from old mathematics documents, the age of the documents increasing the propensity for OCR errors.

PlanetMath (`http://www.planetmath.org`) is also based on LaTeX source. Thus one may assume that all mathematics is given in terms of LaTeX documents. Indeed, the comparison of LaTeX with markup languages mentioned before shows that LaTeX-like dialects are much superior for human modeling of mathematics to XML-based representations. In view of the availability of powerful converters such as those from the ArXMLiv system (`kwarc.eecs.iu-bremen.de/projects/arXMLiv/`), the latter may, however, prove very useful as an intermediate stage for our work. Indeed, ArXMLiv indicates that various tasks involved in "doing mathematics" (e.g., search, navigation, cross-referencing, quality control, user-adaptive presentation, proving, simulation) can be machine-supported, thus relieving working scientists from some routine work, enabling them to concentrate on what humans can still do infinitely better than machines.

We expect that a useful first prototype of our system with limited but meaningful capabilities will be available by the end of the second year, so that external people can start contributing to the success of the project and beyond. Our capacity will be 53.5 man years (50 funded plus 3.5 permanent). Freek Wiedijk's 140 man year estimate for the cost of formalizing 12 textbooks covering the undergraduate curriculum of a typical mathematics study in the spirit of the QED project (see `http://www.cs.ru.nl/~freek/notes/mathstdlib2.pdf`) suggests – in view of our not quite identical goals – that we might contribute about 10–15% (14–21my) towards the completion of the QED project. Since our system will probably make it significantly easier for others to work on formalizing mathematics, our effective contribution to Wiedijk's version of QED might approach perhaps 30%.

Combining our system with the full power of current theorem provers will make the dream of the QED project a reality. Although this is not our immediate goal, the result of the work in the proposed project will be a huge number of exercises or challenges for standard logic solvers. One just needs to write interfaces that translate our proof sketch language into the native language of each logic proof system; and we'll write at least one of these. Then the work of creating the required QED database is naturally split between mathematicians and logicians, and the collection will prove useful immediately, even when only partially verified. This natural division of work is much easier to realize than the full QED project and will lead to substantial time savings for both communities.

MATHRESS will make possible the Automatic Translation of Mathematics. Our system will make important steps towards the automatic translation of mathematical content into other human languages, as well as into versions accessible to visually impaired people. This will be possible because all mathematical text is translated into a concise low-level internal format that gives a complete description of the *meaning* of each phrase. This description can then be faithfully rendered in any language, though often not in a completely isomorphic way, since even in a single language, the same mathematical content can be put into very different words. The automatic translation will be faithful to the contents, not to the style. Our system will be able to translate proofs and formalized statements, but not the background, physical interpretations, motivational comments, historical aspects etc.. This is left to the interaction with the human translator.

The automatic translation facility will have high impact on translating mathematical text into other languages, especially those with few competent mathematics speakers. Since (except in motivating or historically oriented passages) mathematical content is much simpler than arbitrary natural language – and therefore more amenable to automatic translation, our project may also play a pioneering role for automatic translation in other sciences.

MATHRESS will change the way mathematics is done in practice – in the classroom, in research, and in industry. The appearance of powerful calculators changed the way calculation skills are taught and done; something similar will happen with MATHRESS on a higher level. As mathematics permeates science and engineering, from the most elementary aspects to highly complex modeling tasks, the availability of a system like MATHRESS will make it nearly as easy to apply reliably mathematical tools as currently calculators are applied.

As our emphasis is on making MATHRESS able to address large-scale applications, scientists and engineers will directly profit from the ease with which they can do their modeling. It will no longer be necessary to learn specialized languages for solving mathematical problems – the common mathematical language taught anyway to scientists and engineers will provide direct access to the solution facilities. As a result, modeling cycles will become shorter, more complex problems become tractable more easily, and experts can concentrate on the parts where their expertise is most needed.

Human expertise will gradually move away from being able to execute repetitive mathematical thinking activities to being able to evaluate the conditions under which such activities are most usefully employed.

MATHRESS will enable the creation of refereeing tools that authors and/or referees of papers of mathematical and other scientific journals may use to check the formal parts of scientific manuscripts for correctness, thus simplifying the editorial work in the pre-publishing stage.

MATHRESS will also change the way current electronically supported learning of mathematics is done. Current approaches such as web advanced learning technologies (WebALT: `http://webalt.math.helsinki.fi`), advanced calculation and presentation tools for mathematics education (Maths for More: `http://www.mathsformore.com`), and thematic network for the coordination of content enrichment activities in the area of mathematics for e-learning platforms (Joining Educational Mathematics: `http://www.jem-thematic.net`) will need to be complemented by understanding-based approaches, since solving elementary exercises becomes available upon pressing a button.

Part II: Methodology

The proposed project aims at a complex software system whose coherent development requires a careful modular design in order to be completable within the time frame by a group of people working in parallel. Our experience with the COCONUT project (a large software system for global optimization funded by EU between 2000-2004) shows that the work modules need to be chosen in such a way that substantial work can be done in each module already from the beginning. We performed a preliminary analysis during the planning stage and the MoSMath project. This led to a splitting of the workload into five large work packages capturing the scientific part of the work, denoted by capital letters C, T, D, A and I+J, detailed in this part. In addition, there will be two further work packages, one for project management and administration (MA), and one for international cooperation, dissemination and maintenance (CDM), not described here.

Outline of Work Plan. The MATHRESS project is planned to continue far beyond the period of five years for which funding is requested. As traditional in mathematically oriented fundamental research, we shall work on where progress is most apparent, and where it is most needed from the perspective of the whole project. In a weekly seminar, jointly with the already established seminar for the project "A modeling system for mathematics" (MoSMath) funded by the Austrian Science Foundation FWF, we shall discuss progress and literature. We give priority to more basic issues that need to be settled in order that the remainder of the project can proceed smoothly and without later expensive alterations.

Thus the work plan outlined below is only a rough approximation – one possible scenario. Many specific decisions will depend on findings not yet available during the present planning stage. Therefore, while describing below all the work that needs to be done, we specify only which part of the work is expected to be completed or (for long-lasting activities) at least advanced to a useful level within the first year and (with significant uncertainty) within the first five years. At the end of every project year, the project plan will be updated, and made specific for the coming year, thus allowing the necessary adjustments.

Our proposed system will have both a wiki flavor (a huge collaborative workspace for mathematicians), and a butler flavor (where a single scientist can organize his mathematics knowledge, and access it with system help), naturally blended. Users have their own version of the system, based on a downloadable reference version. They can contribute all or part of what they do to a wiki which is left to grow liberally. From the wiki contents, a review panel will select material to include into the next version of the reference version. Thus there will be collaborative, wiki-style organized work by different groups of people in the community: creators of mathematical content (definitions, theorems, proofs), proof checkers who validate the arguments either informally or with the help of formal theorem provers, reviewers/harvestors who evaluate what is there for quality and inclusion into a reference version, and textbook writers who organize the high quality part into well-written electronic textbooks on a subject.

For a successful automatic mathematical research system we need an analysis and efficient formalization of the social process by which human students acquire their communicative capabilities (Subsection C). As mentioned in Part I, dealing with trust is an essential aspect of real-life mathematical work. Modeling and representing trust through an appropriate trust level system will be the subject of Subsection T. Experience with past intelligent systems shows that any efficient complex information system requires the creation of an adapted database system as a prerequisite, in our case a database for storing mathematical theory (Subsection D). Since a main concern is the seamless extension of skills the mathematician is using anyway, we need to develop and adapt algorithms for manipulating mathematical text and database content with a functionality close to that of a working mathematician (Subsection A). Finally, no collaborative software system intended for human use can function without flexible and easy to use human-machine and machine-machine interfaces (Subsection I+J). A number of features naturally belong to several work packages, but are assigned to only one of them in the listing below.

C. Communicative capabilities. An automatic mathematical research system must have similar communication capabilities as a human mathematics student, both with respect to teachers, fellow students, and machines. The social processes by which students acquire and perfect their capabilities must therefore be modeled in detail and implemented. They form the backbone of the interaction abilities of the MATHRESS system with one or several

(human or automatic) agents. The work package naturally splits into a number of partially dependent modules for tasks the system should be able to perform:
C1: (long-lasting, i.e., ≥ 3 years and undertaken in parallel) Read and understand mathematical content.
C2: (long-lasting) Recognize the existence of hidden assumptions, assumed terminology and notation, etc..
C3: (long-lasting) Recognize and use preferred but not binding notational conventions.
C4: (long-lasting) Guess from context the meaning of ambiguous formulations, and check whether the guessed interpretation is consistent with the context.
C5: Ask sensible questions clarifying items that are not well understood or seemingly ambiguous.
C6: Respond sensibly to statements commenting on user (dis)satisfaction.
C7: (long-lasting) Recognize and correct minor spelling errors, and sloppiness in notation or handling degenerate cases.
C8: Recognize the need to learn more about a concept, and to find out where the required information can be found.
C9: (long-lasting) Integrate notation, concepts, algorithms, or proof methods seen repeatedly or formally described into its knowledge base.
C10: Recognize particular styles of presenting mathematics.
C11: Adapt to its partners in communication by selecting an appropriate level of detail or overview.
C12: (long-lasting) Answer questions regarding its understanding and knowledge of mathematical theory.
C13: Assess the quality of an argument or proof and suggest sensible improvements.
C14: Verify proofs, arguments given in a book or by a lecture note, upon request on different levels of formality.
C15: Relate mathematical content to the system's own understanding, and classify the material accordingly.
C16: Formulate meaningful plans for proceeding in a more complex task.
C17: Recognize that variations of the same statement say essentially the same thing, or be able to say how they differ.

The modules C1–C6 are basic and will be in a useful shape by the end of the first year, but with limited functionality in case of C1–C4. The modules C12 and C13 are the main target during the project duration, and depend essentially on C1–C11. We therefore aim at advancing those modules to a useful level by the end of the fifth year or earlier, so that we shall be able to approximate C12 and C13. The modules starting with C14 will be considered as far as time permits but are unlikely to be completed within the funded period of five years.

There is well-known informal work on the art of doing mathematics, proving, and problem solving (e.g., books by Polya) related to this work package. Apart from simple dialog systems for the automatic grading of exercises, we are not aware of significant implementation work related to communicative aspects of mathematics. It will be a main achievement of this project to develop the communicative aspects of doing mathematics from scratch in a formal model and a corresponding implementation.

T: Trust. Human mathematics students take many mathematical facts on trust. This enables them to quickly reach a useful level of expertise. The amount of trust can be adapted locally as required for the task at hand and can be changed in a piecemeal fashion from very informal to completely formalized. An automatic mathematical research system must therefore be able to copy that behavior and properly assess the amount of trust it can put into a web of connected mathematical statements. Modeling and implementing these capabilities is the subject of this work package. It naturally splits into a number of partially dependent modules:
T1: Design a mathematical mechanism to define and inherit trust in a way close to informal practice, and a number of well-defined trust levels.
T2: Recognize which assumptions a given assertion is based on.
T3: Create a trust checker that, given a definition of what is trusted, points out all the untrusted links in existing verifications together with their signatures.
T4: Enable users to specify or even to experiment with the amount of trust they are prepared to put into various sources of information and deduction systems.
T5: (long-lasting) Design a trustable environment in which to execute system manipulations, and create a programming language to specify the algorithms in a way that they can be easily verified automatically for correctness. Whatever cannot be automated will be done interactively. But we strive to keep the automatic part as large as possible.

T6: Enable modular theorem proving at increasingly demanding trust levels.
T7: (long-lasting) Allow users to start with statements and proofs on the standard informal mathematical level, increasing trust selectively at the weakest links in a chain of proofs. In the first years of the project most trust values will probably come from textbook authorities, gradually to be complemented by fully formalized, automatically checkable proofs.
T8: Create a verifiable consistency verification program in which the whole database can be checked for internal consistency of the signatures, by checking the appropriateness of all signatures that are within reach of the system. (For example, it would be unreasonable to require checking whether a reference to Bourbaki is actually in their books. This might change with the availablility of these books in machine-accessible form.)
T9: The documents defining the MATHRESS must ultimately be generated automatically and verified for equivalence with the internal representation by the MATHRESS verbalizer (which should reproduce the document).
T10: (long-lasting) This requires the verifiability (and ultimately, verification) of all algorithms used in the system in a more than heuristic fashion.

The modules T1–T4 are basic and will be in a useful shape by the end of the first year. The module T7 is the main target during the project duration, and depends essentially on T1–T6. We therefore aim at completing the modules T1–T6 by the end of the fifth year or earlier, and having module T7 in a useful shape by then. The modules starting with T8 will be considered as far as time permits but are unlikely to be completed within the funded period of five years.

Except for some (nonautomatic) metadata annotation in XML languages, there exists no trust-related prior work in mathematics. It will be a main achievement of this project to develop the trust aspects of doing mathematics from scratch in a formal model and a corresponding implementation.

D: Database. Human mathematics students acquire with time a large memory of interrelated mathematical facts. This enables them to understand new mathematical texts without more than the usual hints to context, and to apply their understanding to a task at hand. The memory grows naturally with every interaction of a mathematical nature. An automatic mathematical research system therefore needs a database system that is able to store, retrieve and search mathematical content in a well-organized fashion, and increases automatically with continued use. In order to mirror the quality and flexibility of the informal mathematical community, the database must be such that it holds and updates not only user-specific information but also information from a web-based master database holding more authoritative mathematics content approved by a reviewers board (initially consisting of members of our team), as well as information from trusted automatic peers with whose masters a user cooperates.

Since the MATHRESS system shall support the general mathematician as early as possible, one needs to start with the standard undergraduate material. The PI's "Analysis und Lineare Algebra" book contains everything in a uniform style, and its meanings and intentions are completely known to the PI. That this book is in German is an asset, since as a byproduct of the project, there will soon be both a German and an English version of the book, and ultimately versions in many languages. The book just serves as a first test case. All techniques developed to parse and process the book and its contents will become completely general with time, so that they will apply later (with minor adaptations) to any mathematics book in any language. As the project proceeds we aim at incorporating the contents of other books and mathematics repositories if their LaTeX source is available to us.

This work package is concerned with developing such a database system, adapted to the needs of mathematical information. It naturally splits into a number of partially dependent modules. The proposed database system for storing mathematical theory:
D1: allows the systematic storage and retrieval of mathematical concepts, propositions, and proofs;
D2: is fairly independent of notational styles, input language, and underlying mathematical or logical foundation;
D3: (long-lasting) allows for different, also historical approaches to a subject;
D4: has a versioning and backup/restore system for safe upgrading, etc.;
D5: has a standardized transfer language for communicating database content between different database instances and to standard formats like MathML;
D6: gets automatic upgrades from an online reference repository;
D7: (long-lasting) is well-documented to make it easy to use for human readers;

D8: contains external references for background information, authorities, further explanation;
D9: (long-lasting) contains a formal but unverified representation of the mathematical theory at the undergraduate level; at least naive set theory, linear algebra, real analysis, basic algebra, and basic numerical analysis; initially from **ALA**, analysis and linear algebra lecture notes by the PI, **BOYD**, a book on convex optimization by Steve Boyd, and **AS**, the special functions book by Abramowitz and Stegun;
D10: (long-lasting) . . . and of much more theory as the system grows;
D11: (long-lasting) incorporates info from PlanetMath (open source mathematics encyclopedia);
D12: (long-lasting) incorporates info from the Digital Math Archive (published mathematical literature, in DjVu, by Springer);
D13: provides a diagram authoring system that produces high quality mathematical diagrams while preserving the semantics used to create the diagrams;
D14: allows conceptual and structural search;
D15: interfaces seamlessly to Google Scholar and Google Books.

The modules D1–D5 are basic and will be in a useful shape by the end of the first year. The modules D9 and D10 are the main targets during the project duration, and depend essentially on D1–D8 (and many algorithms from work package A). We therefore aim at completing the modules D1–D8 by the end of the fifth year or earlier, and having module D9 and D10 in a useful shape by then. The modules starting with D11 will be considered as far as time permits but are unlikely to be completed within the funded period of five years.

This work package is mainly based on known techniques amply discussed in the literature (database systems in general; Formal Digital Library; Semantic Web: RDF, OWL; work by Solomon Feferman on mathematical graphics support `http://math.stanford.edu/~feferman/papers/Infinite_Diagrams.pdf`). One section (only) of the special functions collection by Abramowitz and Stegun was proposed to be completely formalized by March 2010 by Freek Wiedijk in the project FEAR by March 2010. Prior work by our team includes the COCONUT database library VDBL and the graph template library VGTL, and preliminary work on the ALA lecture notes within the MoSMath project (`http://www.mat.univie.ac.at/~neum/FMathL.html`).

Though time-intensive (partly due to the amount of material to be considered), we expect no special difficulties in this work package. The only significant challenge is to make the structural search sophisticated enough that it can approximate the (much more challenging) conceptual searches a mathematician might want to perform. The recently launched Wolfram Alpha Computational Knowledge Engine takes a first but limited step in this direction.

A: Algorithms. Human mathematics students acquire the ability to read and understand mathematical language, and to relate the contents to their mental activities. An automatic mathematical research student mimics this by well-defined algorithms simulating these tasks. This work package is about modeling the human behavior and translating it into the design and implementation of algorithms able to imitate the human aspects on an automatic level. The work package naturally splits into a number of partially dependent modules that interface or develop algorithms for:
A1: (long-lasting) Reading and interpreting mathematical text specified in MATHRESS format, designed as a LaTeX-like language as close to natural mathematical language as possible. This requires the design of an internal MATHRESS representation of mathematical content.
A2: (long-lasting) Potential verification of the input process. Context-free grammars must be definable and interpretable. Since mathematical grammar is ambiguous and incremental, an easily verifiable incremental LR parser is needed, and a formal specification of the parser to ensure correct rendering of natural text and exact translation to/from the database.
A3: (long-lasting) Displaying contents in natural mathematical language, if possible in different styles and/or languages.
A4: Hierarchical arrangement of mathematical material in the form of blocks related by a DAG structure, in ways comprehensible by humans.
A5: (long-lasting) Verifying the semantical reasonableness of mathematical content using heuristics and interactive questioning.
A6: (long-lasting) Heuristic evaluation of the quality of mathematical content w.r.t. the given goals; this requires preliminary answers to what constitutes elegance, readability, clarity, beauty, and depth.

A7: (long-lasting) Verifying the logical correctness of proofs given the validity of auxiliary results used, by interfacing with existing theorem provers.
A8: Automatic calculation of routine computations.
A9: (long-lasting) Heuristic decision making for how to reach assigned goals.
A10: Finding counterexamples of (from a human point of view) obviously wrong statements, pointing out obvious gaps and errors in proofs.
A11: (long-lasting) Adapting mathematical content to a personal notation style.
A12: Automatic recognition and formation of special cases and generalizations.
A13: Display formal text in a way accessible to visually or auditorily impaired people.
A14: Comparing different sections of mathematical content with respect to structural similarity and common information.
A15: Meaningful handling of mathematically similar fragments of theories, to be able to do proof by analogy, to recognize canonical constructions such as those abstracted by category theory, to select from variations of a result a "canonical" version.
A16: Restructuring mathematical content according to specific goals.
A17: Waking and sleeping phases for highly attentive foreground work and reorganizing background work, respectively.

In each case, the algorithms must be documented, giving a precise specification, a translation of this specification into human-readable terms, and (sooner or later, probably not within the first 5 years) a formal verification that the implementation satisfies the specification.

The modules A1–A4 are basic and will be in a useful shape by the end of the first year; A1 in a limited form. They overlap with some goals in the MoSMath project that the PI's research team began in May 2008; thus a substantial synergetic effect will be achieved. The modules A1-A9 shall be completed by the end of the fifth year or earlier, and the modules A10–A13 shall by then be in a reasonably good shape. The modules starting with A14 will be considered as far as time permits but are unlikely to be completed within the funded period of five years.

This work package is mainly based on known techniques amply discussed in the literature (extensive parser-generator literature, Naproche, Attempto, the grammatical framework GF, and many logical frameworks, theorem provers, and problem solving environments). Closest to our needs for easy interfacing are (in different directions): Flex/Bison parser-generator, Claus Zinn's thesis, Naproche, Isabelle/Isar, Mathematica, MathWiki, and GF for multilingual support. Their adaptation to the special demands of informal mathematics is sometimes challenging if the task is to be performed at a high level of quality. Many of the tasks (e.g., fully formalized theorem proving) can be delegated to existing specialized systems by writing appropriate interfaces. We do not intend to advance the state of the art in these specialized directions, but simply make existing implementations available to the automatic system.

I+J: Human-machine (I) and machine-machine (J) interfaces. An automatic mathematical research system is useless for the working scientist unless it can easily communicate with humans. As any human mathematics student, it also needs the ability to communicate with and learn from peers – other software systems with similar or complementary capabilities. The interfaces needed to make this possible are the subject of this work package. It naturally splits into a number of partially dependent modules. The human-machine interface consists of
I1: a facility to easily inspect and edit the database;
I2: graph-based techniques to understand the interdependence of the content of the database;
I3: (long-lasting) a dialog system for communication with the user; both the system and the user must be able to ask and respond to simple questions by the other side;
I4: web-based interaction tools to allow remote users to contribute to the growth and quality of the database contents;
I5: (long-lasting) a TEXmacs-like editor for creating mathematical content, for editing mathematical manuscripts for publications, in some format designed as a variant of LATEX;
I6: automatic synchronization tools between different instances of the database;
I7: (long-lasting) an advanced dialog system for communication with the user; both the system and the user are able to ask and respond to questions that sound like normal questions for a human mathematics student;

I8: (long-lasting) a batch mode for the automatic (rough) interpretation of mathematical textbooks, and their incorporation into the database;
I9: review mechanisms for contribution to the MATHRESS by third parties;
I10: graphical tools for mathematical modeling of applications.

Machine-machine interfaces consist among others of
J1: (long-lasting) a LaTeX to MATHRESS translator;
J2: an XML to MATHRESS translator;
J3: interfaces with decisions trees for mathematical software;
J4: (long-lasting) translators of proofs from known solvers to the internal MATHRESS format;
J5: (long-lasting) translators of theory databases from current proof assistants to the MATHRESS format;
J6: a distributed environment for the collaboration of several automatic students via the web.

The modules I1–I5 and J1–J2 are basic and will be in a useful shape (some in a first, limited form only) by the end of the first year, and completed by the end of year 5 or earlier. J1–J2 overlap with goals in the MoSMath project that the PI's research team began in May 2008; thus a substantial synergetic effect will be achieved. The modules I6–I9 and J3–J6 will by then be in a fairly useful shape, although I10 is probably beyond the 5 years horizon. Most closely related to the J tasks are the existing systems ArXMLiv and Naproche, as well as the now dead MONET project, which we intend to exploit.

This work package is mainly based on known techniques amply discussed in the literature; however, the amount of work to adapt it to a system easy to use by the average scientist is significant.

References

[1] C.A. Addison, W.H. Enright, P.W. Gaffney, I. Gladwell, and P.M. Hanson. Algorithm 687: a decision tree for the numerical solution of initial value ordinary differential equations. *ACM Trans. Math. Software*, 17:1–10, 1991.

[2] P. Bunus. A simulation and decision framework for selection of numerical solvers in scientific computing. In *Proc. 39th ann. Symp. Simulation, IEEE Computer Society, Washington*, pages 178–187, 2006.

[3] L. Carlson, J. Saludes, and A. Strotmann. State of the art in multilingual and multicultural creation of digital mathematical content, 2005. Web Advanced Learning Technologies, Manuscript.

[4] E.N. Houstis, A.C. Catlin, N. Dhanjani, and J.R. Rice. MyPYTHIA: a recommendation portal for scientific software and services. *Concurrency Computat.: Pract. Exper.*, 14:1481–1505, 2002.

[5] ISO. International standard ISO/IEC 13568:2002 information technology – Z formal specification notation – syntax, type system and semantics, 2002.

[6] J. Kallrath. *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis*. Vieweg, 2002.

[7] F. Kamareddine. Motivations for MathLang, 2005. Slides.

[8] F. Kamareddine, M. Maarek, K. Retel, and J.B. Wells. Gradual computerisation/formalisation of mathematical texts into Mizar, 2004. Manuscript.

[9] F. Kamareddine, M. Maarek, and J.B. Wells. *Flexible encoding of mathematics on the computer*, pages 160–174. Springer, Berlin, 2004.

[10] F. Kamareddine, M. Maarek, and J.B. Wells. MathLang: An experience driven language of mathematics. *Electronic Notes in Theoretical Computer Science*, 93C:123–145, 2004.

[11] S. Maaruster, V. Negru, D. Petcu, and C. Sandru. Intelligent front-end for solving nonlinear systems of differential and algebraic equations. *J. Math. Sci.*, 108:1139–1151, 2002.

[12] N. Ramakrishnan and C.J. Ribbens. Mining and visualizing recommendation spaces for elliptic pdes with continuous attributes. *ACM Trans. Math. Software*, 26:254–273, 2000.

[13] N. Ramakrishnan, J.R. Rice, and E.N. Houstis. GAUSS: an online algorithm selection system for numerical quadrature. *Adv. Engin. Software*, 33:27–36, 2002.

[14] D.L. Simon. *Checking Number Theory Proofs in Natural Language*. PhD thesis, Univ. of Texas, Austin, 1990.

[15] P. van Tilburg. Exploring the core of MathLang, 2006. Manuscript.

[16] F. Wiedijk. Formalizing 100 theorems, 2007. WWW-Document.

[17] F. Wiedijk. The seventeen provers of the world, 2007. WWW-Document.

[18] C. Zinn. *Understanding Informal Mathematical Discourse*. PhD thesis, Univ. Erlangen, 2004.