SErAPIS : A Concept-Oriented Search Engine for the Isabelle Libraries Based on Natural Language *

Yiannos Stathopoulos
Angeliki Koutsoukou-Argyraki and
Lawrence C. Paulson

Department of Computer Science and Technology, University of Cambridge, UK.
{yas23, ak2110, lp15}@cam.ac.uk

Abstract. We introduce SErAPIS, a work-in-progress search engine for the Isabelle libraries. SErAPIS uses natural language and an offline index, meaning that search is done in all Isabelle libraries and there is no restriction to the libraries and theories that are loaded in the active session. We describe the SErAPIS pipeline that indexes Isabelle facts by associating them to words and concepts (special phrases that refer to mathematical structures, objects and ideas) in Wikipedia mathematics articles. We also describe three models for extracting textual descriptions of facts from these articles and present a preliminary experiment with four retrieval models based on our indexing methods. We are planning several further developments to the engine as well as an extensive evaluation using online user data.

Keywords: search · information retrieval · natural language processing · proof assistants · Isabelle · libraries of formalised mathematics.

1 Introduction

Isabelle users can benefit from easy and real-time access to formalised material in the libraries while constructing their proofs. Current approaches to indexing and searching Isabelle theories fall into one of two categories. The first approach involves pattern matching of strings against names of objects in loaded libraries. For example, Isabelle's `find_theorems` retrieves Isabelle objects using symbolic pattern matching on objects in libraries loaded in the active session (i.e., online search). However, inexperienced users might have an idea of what facts they need to complete their proof but not enough knowledge of the Isabelle library organisation and naming conventions to construct effective queries for `find_theorems` [5]. The second approach involves abstracting the mathematical knowledge in Isabelle's libraries using a formal meta language, such as MMT [10, 7].

SErAPIS (Search Engine by the Alexandria Project [9] for ISabelle) is a new, work-in-progress search engine based on natural language descriptions of facts in the libraries. It is not a formula search engine per se, but a fact search engine. It makes use of words to model the topic of each fact and mathematical concepts

* Supported by the ERC Advanced Grant ALEXANDRIA (Project 742178).

in natural language: phrases that refer to mathematical objects, structures and ideas. SErAPIS is an offline search engine – it searches the complete Isabelle libraries independently of which libraries or theories are loaded in the active session using a pre-computed index. This is helpful as users do not always know where the needed material may be located. Searching Isabelle using SErAPIS is done by entering keywords in a search box and, optionally, selecting mathematical concepts from a list through a simple user interface (Figure 1). In the next section we introduce ideas from text-based Information Retrieval (IR) and discuss relevant work. In Section 3 we describe the SErAPIS indexing pipeline and explain how keywords and concept phrases are extracted for each fact. A preliminary experiment using SeRAPIS is presented in Section 4.

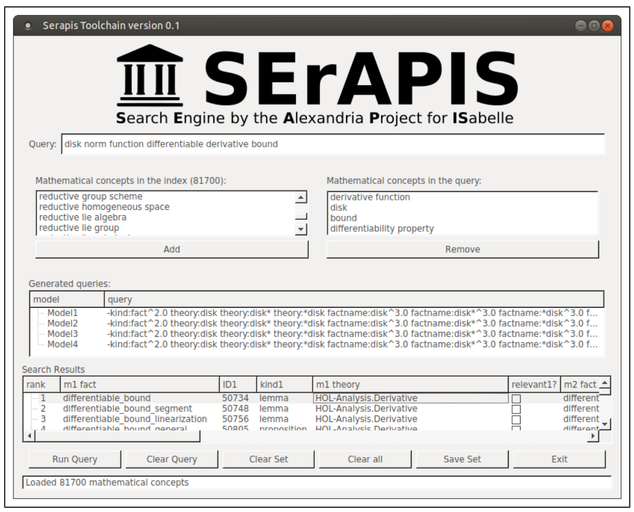

Fig. 1. The SErAPIS desktop user interface for collecting relevance judgements.

2 Background and Related Work

Mathematical knowledge management (MKM) and mathematical IR (MIR) are active fields of related research. MIR focuses on retrieving mathematical documents using textual queries and formulae. Many retrieval systems supporting formula search, such as MCAT[6], MathWebSearch [3] and Tangent [8] have emerged from work in this field. Recent efforts in managing formal collections of mathematical knowledge have resulted in the OMDoc semantic markup language [4]; and the MMT system and language [10, 1] which abstract mathematical knowledge into a formal meta-logic (which also supports indexing and search of the Isabelle libraries).

Modern IR systems for natural language employ the *bag-of-words paradigm* and the *Vector-Space Model* (VSM). The bag-of-words paradigm assumes that words are independent discriminators of thematic similarity. The VSM models documents and queries as term vectors: n-dimensional vectors where each word in the dictionary is represented by a weighted vector component. Similarity between documents and queries is computed as the cosine similarity between the document and query vectors. The values of the components in these vectors are typically determined using the TF-IDF [11] weighting scheme. TF-IDF is proportional to the term frequency (TF, number of occurrences of a word in a document) and inverse document frequency (IDF, the number of documents a word occurs in a document collection such as Wikipedia) to measure the discriminating ability of a word.

Our approach is different to that of the OMDoc and MMT initiatives since our goal is to make the Isabelle libraries available for search through a simple, natural language interface akin to Google. We also adopt the idea of "mathematical types" from text-based mathematical IR [13, 12] and use the Cambridge Dictionary of Mathematical Types (CDMT) as a source of phrases that refer to mathematical objects, structures and ideas [12].

In the work by Condoluci et al. [2], an upper library ontology (ULO) for mathematical knowledge consisting of information about facts across libraries is introduced. In contrast, SErAPIS is a concept-oriented search engine for facts based on natural language (i.e., akin to Google search) for Isabelle in particular.

3 Building a Concept-Oriented Search Engine for Isabelle

At the core of SErAPIS is a Lucene index constructed by the four-step pipeline shown in Figure 2. The first step in the pipeline is to obtain the Prover IDE (PIDE) [14, 15] markup of all theories in the Isabelle distribution using `isabelle dump` [16].

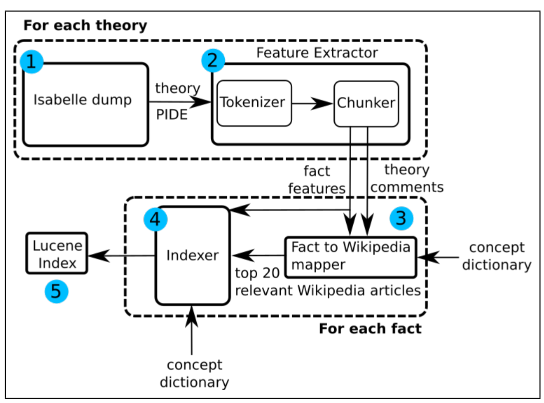

Fig. 2. The SErAPIS index construction pipeline.

By adopting PIDE markup as our input we are able to index new releases of the Isabelle libraries without modifying our pipeline. More importantly, PIDE markup allows us to extract information from theories without having to parse them, a non-trivial task because Isabelle's syntax is ambiguous and valid parse trees can only be selected after type-checking.

The second step in our pipeline is feature extraction. Our feature extractor pre-processes the PIDE stream into a sequence of tokens, each token being a tree representing Isabelle inner or outer syntax commands and Isar constructs. The chunker then groups the tokens into chunks, using a recursive descent shallow parser, that represent larger constructs such as theorems, lemmata, propositions and definitions. Example output of our tokeniser for the lemma `finite_B` in `HOL-Number_Theory.Gauss` is presented in Figure 3 in Appendix A.

Our feature extractor collects 20 features (some multi-dimensional vectors) from `theorem`, `lemma`, `corollary`, `proposition`, `axiom` and `definition` chunks which we list in full in Table 2 in Appendix B. Two important features for generating the SErAPIS index are "comments" (comments above a fact in the theory file) and "incomments" (comments that appear in the fact's body). We also collect features intended for future extensions of SErAPIS, such as the occurrences of symbols in a fact's body (e.g., in the statement and proof of a lemma) and features that represent outgoing references to other facts. The feature extractor produces two tables: (i) facts and their features and (ii) aggregated comments for each theory.

Next, we map each fact to a set of 20 thematically related mathematical Wikipedia articles as follows. We produce a supporting Lucene index [1] for mathematical Wikipedia articles that stores their (a) title, (b) body text and (c) two fields that list those mathematical concepts from the CDMT that occur in the title and body. The Wikipedia mapper identifies the top 20 most relevant articles to each fact by querying the support index using keywords obtained from the fact's name, comment-related features about the fact (comments/incomments) and the comments present in its source theory. The identification of mathematical concepts in the top 20 results is achieved by applying multi-pattern string matching of all concepts in our dictionary to the body and title of the Wikipedia articles.

The SErAPIS indexer uses the fact feature table, the concept dictionary and the set of 20 most relevant articles to find keywords and concepts that describe each fact. Some features from Table 2 are propagated into the index for use by future extensions. The fact inherits words and concepts in the top 20 articles using one of three different methods that we developed for aggregating the linguistic terms and mathematical concept phrases from the articles to generate descriptions for each fact.

Method 1. A textual description for each fact is produced by summing up the term vectors (dictionary of words and their frequency) of the 20 Wikipedia articles that are most relevant to that fact. Similarly, we sum up the concept vectors of the articles to produce a bag of concepts presumed to be relevant to the fact.

Method 2. A textual description for each fact is produced by selecting the 100 most important terms and concepts from its associated Wikipedia article set. We use the TF-IDF statistic [11] to rank terms and concepts by their discriminating ability as observed in the subset of Wikipedia about mathematics.

[1] a searchable data structure produced by the Lucene information retrieval software library.

Method 3. The textual description of a fact is produced by finding the terms and concepts that maximally overlap the set of relevant articles. This is done by first producing a 20×20 matrix whose cells contain the Jaccard coefficient between pairs of articles. At first, the cell with the highest Jaccard in the table is added to the output list. Subsequent entries are added iteratively by finding the cell with the highest Jaccard in the same row or column as the current cell. This step is repeated until the list contains $K = 40$ pairs. Finally, the list is folded by taking the word (or concept) set pairs and intersecting them. This produces the final word (or concept) set used to produce the description for the fact.

4 Preliminary Experiment

We prepared 25 search queries (Table 3 in Appendix C) for an internal preliminary test designed to simulate a user describing a fact (whose exact name is unknown) to be retrieved using words and concepts. First, we came up with an information need and an example fact that would satisfy it. Then, we came up with words that describe this fact but do not exactly match its name [2]. Concept phrases for each query are proposed by the interface in the form of a list and users can select one or more concept phrases manually, i.e., concepts were created by selecting phrases from the CDMT that are topically related to and are constructed entirely by one or more of the input query words. An input query generates four searches, which correspond to retrieval models. Models 1–3 use concepts and words in the query, which are matched against the index fields generated by the indexer for the corresponding methods (Section 3). A fourth model is our word-only retrieval baseline and does not use concept phrases. We used SErAPIS (Figure 1) to pool results from the four models and judged the top-20 results for each model for relevance (binary). A fact is judged as relevant if it sufficiently satisfies the description by the query words and concepts. In particular, in the cases where the search keywords consisted of a main notion and a secondary notion, wherever a result involved the main notion but not the secondary, it was judged still as relevant. Wherever the main notion was absent in the search result it was judged as non-relevant. We use mean average precision (MAP) to measure model retrieval performance and the powerful non-parametric permutation test (paired) to compare model runs.

	Model 1	Model 2	Model 3	Model 4
MAP	.775	.659	.731	.688
Model 1	-	$>$	$>$	$>$
Model 2	$<$	-	$\approx$	$\approx$
Model 3	$<$	$\approx$	-	$\approx$
Model 4	$<$	$\approx$	$\approx$	-

Table 1. Model retrieval efficiency comparison.

Table 1 shows that model 1 significantly outperformed (at $\alpha = 0.05$) the other models. This suggests that summing up the term and concept vectors

[2] e.g., we used the words "summability", "zero", "criterion" instead of "summable", "null", "test" which many related facts contain in their names.

from the top-20 Wikipedia articles is a promising method. Model 3 performs well but is not as good. Retrieval models that use concept phrases in addition to words in the query outperformed model 4 which does not, with the exception of model 2 (descriptions based on TF-IDF discriminating ability). An extensive direct comparison of these models to `find_theorems` is infeasible because the performance of the latter depends on the libraries loaded by the user in the active session at query time. Three characteristic examples where SErAPIS produces far more results than `find_theorems`[3] (presumably because of its reliance on symbolic pattern matching) are in Appendix D.

5 Conclusion and Future work

In this paper we introduced SErAPIS, described its indexing pipeline and presented a preliminary experiment that suggests that its text-based approaches can be useful. We intend to add the Archive of Formal Proofs to the index and to perform an extensive evaluation, using data collected from Isabelle users through an online version of SErAPIS. We also plan to experiment with other description-generating methods, such as taking advantage of the referential graph between facts. At this stage we already have a working prototype of the online version of SErAPIS, which can be found at http://behemoth.cl.cam.ac.uk/serapis/search.php (please see the user guide on the page). We are in the process of developing this prototype into a platform for collecting user queries for large scale evaluation which will be available at behemoth.cl.cam.ac.uk/serapis.

References

1. Betzendahl, J., Kohlhase, M.: Translating the IMPS Theory Library to MMT/OMDoc. In: Intelligent Computer Mathematics - 11th International Conference, CICM 2018, Hagenberg, Austria, August 13-17, 2018, Proceedings. pp. 7–22 (2018)
2. Condoluci, A., Kohlhase, M., Müller, D., Rabe, F., Sacerdoti Coen, C., Wenzel, M.: Relational data across mathematical libraries. In: Kaliszyk, C., Brady, E., Kohlhase, A., Sacerdoti Coen, C. (eds.) Intelligent Computer Mathematics. pp. 61–76. Springer International Publishing, Cham (2019)
3. Hambasan, R., Kohlhase, M., Prodescu, C.: MathWebSearch at NTCIR-11. In: Proceedings of the 11th NTCIR Conference on Evaluation of Information Access Technologies, NTCIR-11, National Center of Sciences, Tokyo, Japan, December 9-12, 2014 (2014), http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings11/pdf/NTCIR/Math-2/05-NTCIR11-MATH-HambasanR.pdf
4. Kohlhase, M.: OMDoc - An Open Markup Format for Mathematical Documents [version 1.2], Lecture Notes in Computer Science, vol. 4180. Springer (2006). https://doi.org/10.1007/11826095, https://doi.org/10.1007/11826095
5. Koutsoukou-Argyraki, A.: Formalising Mathematics – in Praxis ; A Mathematician's First Experiences with Isabelle/HOL and the Why and How of Getting Started (submitted preprint) (07 2019)

[3] the words "infimum"and "supremum" return no results having loaded libraries in which SErAPIS does find many results.

6. Kristianto, G.Y., Ho, F., Aizawa, A.: The MCAT Math Retrieval System for NTCIR-11 math track. In: In Proc. NTCIR Workshop 11 Meeting (2014)
7. Müller, D., Gauthier, T., Kaliszyk, C., Kohlhase, M., Rabe, F.: Classification of Alignments Between Concepts of Formal Mathematical Systems. pp. 83–98 (06 2017). https://doi.org/10.1007/978-3-319-62075-6_7
8. Pattaniyil, N., Zanibbi, R.: Combining TF-IDF Text Retrieval with an Inverted Index over Symbol Pairs in Math Expressions: The Tangent Math Search Engine at NTCIR 2014. In: Proceedings of the 11th NTCIR Conference on Evaluation of Information Access Technologies, NTCIR-11, National Center of Sciences, Tokyo, Japan, December 9-12, 2014 (2014), http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings11/pdf/NTCIR/Math-2/08-NTCIR11-MATH-PattaniyilN.pdf
9. Paulson, L.: ALEXANDRIA: Large-Scale Formal Proof for the Working Mathematician. https://www.cl.cam.ac.uk/ lp15/Grants/Alexandria/
10. Rabe, F.: The MMT Language and System
11. Spärck-Jones, K.: A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation **28**, 11–21 (1972)
12. Stathopoulos, Y., Baker, S., Rei, M., Teufel, S.: Variable Typing: Assigning Meaning to Variables in Mathematical Text. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). pp. 303–312. Association for Computational Linguistics, New Orleans, Louisiana (Jun 2018). https://doi.org/10.18653/v1/N18-1028, https://www.aclweb.org/anthology/N18-1028
13. Stathopoulos, Y., Teufel, S.: Mathematical Information Retrieval based on Type Embeddings and Query Expansion. In: Proceedings of the 26th International Conference on Computational Linguistics, Coling 2016, December 11-16, 2016, Osaka, Japan. pp. 334–340 (2016), http://aclweb.org/anthology/P/P15/P15-2055.pdf
14. Wenzel, M.: PIDE as front-end technology for Coq. CoRR **abs/1304.6626** (2013), http://arxiv.org/abs/1304.6626
15. Wenzel, M.: Isabelle/PIDE after 10 years of development. In: UITP 2018 (International Workshop on User Interfaces for Theorem Provers 2018). (2018), https://sketis.net/wp-content/uploads/2018/08/isabelle-pide-uitp2018.pdf, https://sketis.net/wp-content/uploads/2018/08/isabelle-pide-uitp2018.pdf
16. Wenzel, M.: The Isabelle/Isar Reference Manual (2019), https://isabelle.in.tum.de/doc/isar-ref.pdf, https://isabelle.in.tum.de/doc/isar-ref.pdf

A Pipeline Chunker Example

<table>
<tr><th>Isabelle</th><th>Chunked Tokens</th></tr>
<tr><td>lemma finite_B:
    "finite B"
    by (auto simp add:
        B_def finite_A)</td><td><command 1> 'lemma'
  <text >'lemma'
<fact local.finite_B> 'finite_B'
<delimiter> ':'
<proposition delimited=true>
  <text >'"'
  <function> finite
        'Int.int' => 'Set.set'
                => 'HOL.bool'
  <function> B
        'Int.int' => 'Set.set'
  <text >'"'
<command 1> 'by'
  <text >'by'
<method meta=null>
  <delimiter> '('
  <operator operator> 'auto'
  <command 4 method_modifier> 'simp'
  <command 4 method_modifier> 'add'
  <delimiter> ':'
  <fact local.B_def> 'B_def'
  <fact local.finite_A> 'finite_A'
  <delimiter> ')'</td></tr>
</table>

Fig. 3. Lemma "finite_B" in HOL-Number_Theory.Gauss (left) and the output of our chunker for this lemma (right).

B Extracted Features for Isabelle Facts

General Features

	Feature	**Kind**	**Description**
1	name	String	The name of the fact
2	kind	String	The kind of the fact: theorem, lemma, definition or axiom.
3	theory_key	String	Identifier for the source theory in Library_Theory format.
4	theory_name	String	The name of the source theory, produced from its filename.
5	comments	Text	Comments above the fact in the theory file.
6	incomments	Text	Comments appearing inside the fact's body.

Fact body Features

	Feature	**Kind**	**Description**
7	commandvec	Vector	Inner syntax of Isar commands and their frequency.
8	opvec	Vector	Operators that appear in the body of the fact and their frequency.
9	constvec	Vector	Constants that appear in the body of the fact and their frequency.
10	refvec	Edges	Other facts referenced in the body of the fact and the frequency of their evocation.
11	typevec	Vector	Isabelle types used in the body of the fact and their frequency.

Proof Block Features

12	**Feature**	**Kind**	**Description**
13	proofblocks	Integer	The number of "proof" blocks in the fact's body.
14	byblocks	Integer	The number of "by" blocks in the fact's body.
15	proof_commandvec	Vector	Commands used in the proof block of the fact and the frequency of their evokation.
16	proof_methodvec	Vector	Methods used with the "by" command and their frequency.
17	proof_opvec	Vector	Operators used in the proof block and their frequency.
18	proof_constvec	Vector	Constants used in the proof block and their frequency.
19	proof_refvec	Edges	Other facts referenced in the proof block and the frequecy by which they are used.
20	proof_typevec	Vector	Isabelle types used in the proof of the fact and their frequency.

Table 2. Complete set of features recorded for every fact in the Isabelle libraries.

C Search Queries

ID	Query Keywords	Query Concepts	# Relevant Facts
1	disk, norm, function, differentiable, derivative, bound	"derivative function", "disk", "bound", "differentiability property"	35
2	borel, measure, basis, box	"borel measure", "basis"	20
3	summability, criterion, test, norm, less, comparison	"test", "comparison", "summability condition", "summability", "norm"	22
4	multiply, less, positive	"multiply element", "positive number", "multiply"	6
5	summation, test, geometric, series	"summation", "summability condition", "geometric series", "summable series"	22
6	norm, limit, summability, less, criterion, test, comparison	"comparison", "summability condition", "summability", "limit", "norm"	16
7	series, summability, ratio, test	"series", "summability condition", "summability", "ratio test"	26
8	summability, criterion, index, shift	"index change", "shift", "summability condition", "summability"	9
9	limit, zero, comparison, less	"comparison", "zero", 'limit'	39
10	summation, subtract, minus	"summation", "minus operation", "subtract operation"	45
11	summation, telescoping, index, shift	"summation", "index change", "sum", "telescoping series"	19
12	simplification, rules, division	"simplification rule", "division"	34
13	equivalent definition derivative	"derivative", "equivalent definition"	26
14	set, membership, ordering, less, greater	"ordering", "set membership relation"	33
15	neighborhood, filter, derivative, metric	"filter", "derivative", "metric", "neighborhood"	28
16	caratheodory, characterisation, derivative, continuous	"characterisation theorem", "continuity", "derivative"	34
17	vector, space, linear, components	"linear component", "vector space"	23
18	homotopy, of, maps, product, topology, homotopic	"homotopic map", "product topology"	16
19	triple, curve, theorem, path	"path", "curve"	45
20	Lebesgue, measure, monotonic, function	"monotonic function", "lebesgue measure"	21
21	harmonic, numbers	"harmonic number"	28
22	winding, number, simple, closed, path	"closed path", "winding number"	21
23	absolutely, convergent, product	"convergent product"	21
24	well, order, embedding, reflexive	"well order", "reflexive property", "embedding"	40
25	polynomial, ring, irreducibility	"irreducibility criterion", "polynomial ring"	20

Table 3. Complete query set for the SErAPIS evaluation. The top-20 results for each method were judged and relevant facts were pooled from all methods.

D Infimum, Supremum and Harmonic Numbers Results

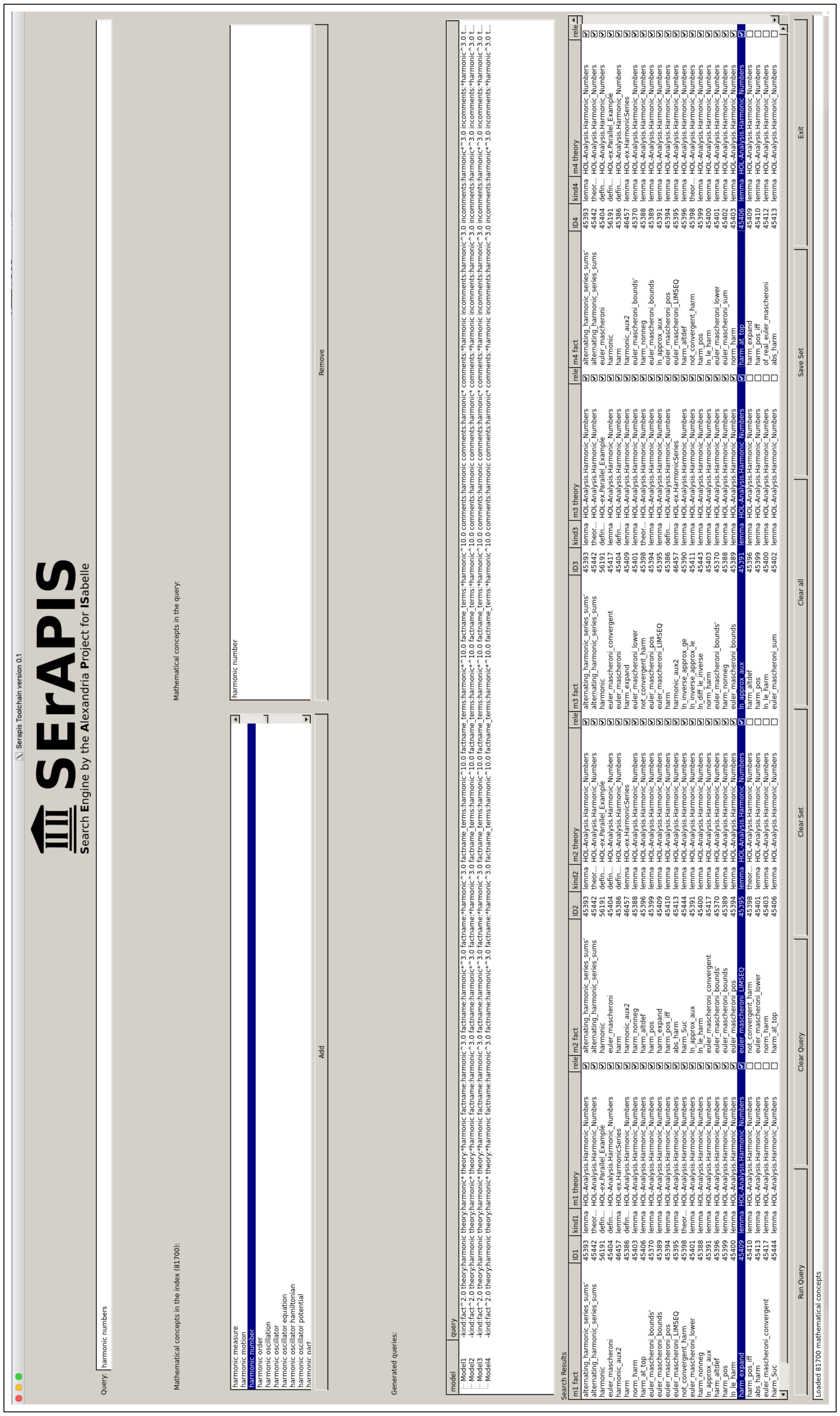

Fig. 4. SErAPIS results for “harmonic number”.

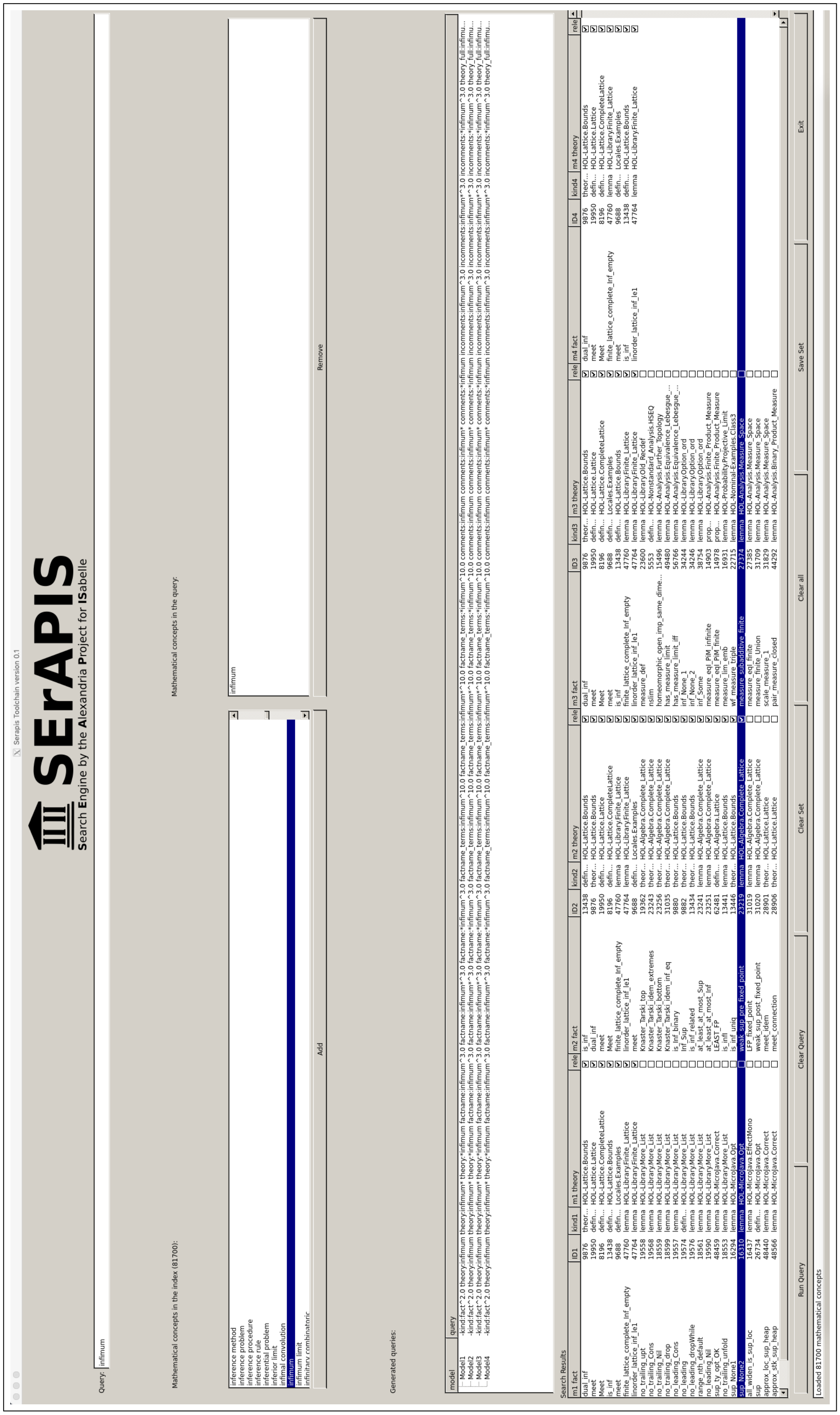

Fig. 5. SErAPIS results for “infimum”.

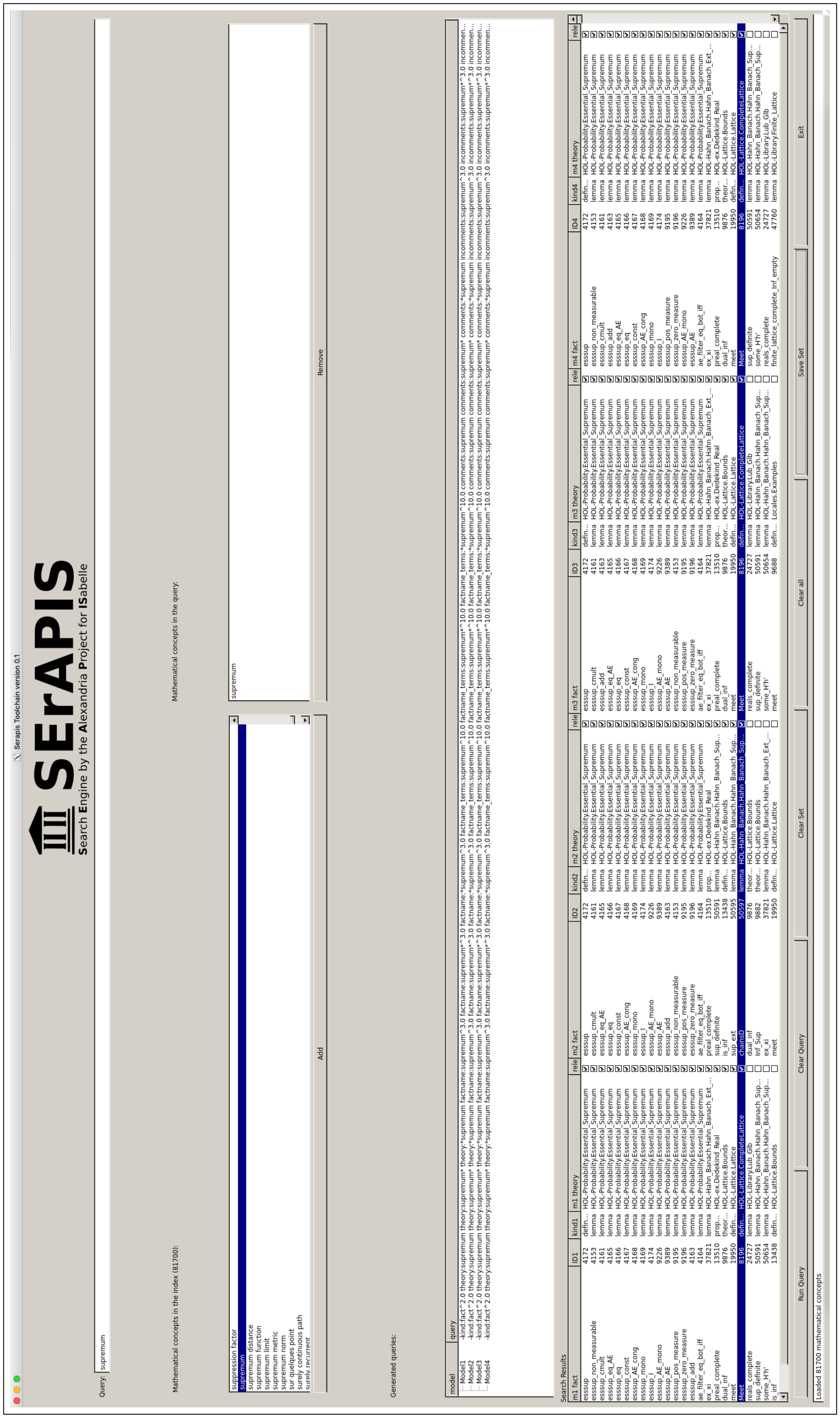

Fig. 6. SErAPIS results for "supremum".