— SiSsI Blue Note[*] —
TeMaSearch: Combining MathWebSearch with Text Search

Michael Kohlhase
Computer Science, Jacobs University Bremen
http://kwarc.info/kohlhase

August 18, 2011

Abstract

This note sketches an idea for combining MathWebSearch formula search with bag-of-words search technologies to achive full-text search (formulae and text) for mathematical documents. In this algorithm, formula search is used as a query extension for conventional search. The the TeMaSearch algorithm combines the advantages of the both components: formula search with wildcards from MathWebSearch and search ranking, scoring from the bag-of-words engine.

1 Introduction

MathWebSearch is an unification-based search engine for mathematical formulae [**KohSuc:asemf06**; **MathWebSearch:online**; **ProKoh:mwssofse12**]. It would be great, if we could combine this functionality with regular text search capabilities. For **verbalization-based formula search engines** (i.e. engines that transform formulae into "words" by verbalizing them and then use a conventional search engine to index them and process queries) such as [**MilYou:tadlmf02**; **MunMin:MathFind06**; **LibMel:marmca06:biblatex**; **MisGal:egoMath11**], this functionality is immediate by design, but for MathWebSearch, which uses a custom indexing method to offer unification-based querying just inserting words into the index is not an option.

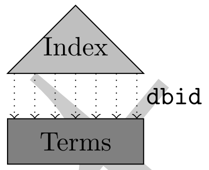
A combination of MathWebSearch with text search has been attempted before [**Anca:matesearch07**], but the method used there has not proven fully satisfactory: In his MaTeSearch system Ştefan Anca separated search for words (using nutch [**nutch:on**]) and formulae (using MathWebSearch) and then integrated the results via a (weighted) intersection. The problem in this approach was that in the absence of robust and meaningful ranking and scoring algorithms, weighted intersection led to somewhat arbitrary results.

In this note we present an alternative combination, where we essentially use MathWebSearch for query expansion in a conventional search engine. We observe that MathWebSearch is similar to a one-word IR algorithm, except that unification directly matches one search term against lots of search terms. This suggests that we can combine unification indexing with the vector space model for a "bag-of-formulae" (instead of standard IR's "bag-of-words") method.

2 The Combination

[*]Inspired by the "blue book" in Alan Bundy's group at the University of Edinburgh, SiSsI blue notes, are documents used for fixing and discussing ϵ-baked ideas in projects by the SiSsI group (see http://kwarc.info). Unless specified otherwise, they are for project-internal discussions only. Please only distribute outside the SiSsI group after consultation with the author.

To understand the proposed algorithm, consider the way MathWebSearch builds an index. Every formula φ is decomposed into a substitution sequence $\widetilde{\varphi}$ which is then inserted into a top-down substitution tree (the light gray triangular structure in the figure on the right), every leaf in the index tree is assigned an identifier `dbid`. Then φ is inserted into a term databse (the darker gray box on the right) indexed with the `dbid` of $\widetilde{\varphi}$. Note that multiple occurrences of formulae in a document share the same `dbid`. The main idea of the TeMaSearch is to "verbalize" formulae as their `dbid` and use MathWebSearch as a form of query expansion.



The TeMaSearch algorithm has two parts: one for indexing a document and one for querying.

At Indexing time (i.e. when we index a math document D),
I1 insert the formulae into the MathWebSearch index, remember `dbid`
I2 replace all formulae in D with their `dbid`[1] to get a document D'
I3 index D' in a bag-of-words search engine $\mathcal{S}$, e.g. Apache Solr [**Solr:on**]

An interesting question to empirically test here is how the data consumption of $\mathcal{S}$ grows with the influx of new "words" that are verbalized formulae ([**ProKoh:mwssofse12**] estimates the non-trivial, indexable formula occurrences of the arXiv corpus at 0.6 billion).

At query time
Q1 query Q consists of a set Q_f of formulae and a set Q_w of words.
Q2 run Q_f through MathWebSearch to get set I_f of matching `dbid`s.
Q3 run $Q' = Q_w + I_f$ through $\mathcal{S}$ to get a set R of document fragments URIs.
Q4 we return R together with the fragments of D they point to.
Note that all the wildcard handling in formulae is done by the unification algorithm at the heard of MathWebSearch and all advanced search features (wildcards in words, logical operators, etc.) can be inherited from $\mathcal{S}$.

3 Conclusion

We have presented an idea for combining unification-based formula search with text search. This idea seems to have advantages over an intersection-based solution tried earlier. It even allows to inherit the ranking mechanisms from the underlying text search engine.

This idea has been presented to the German DB/IR workshop in Berlin in March 2012 and was encouraged.

[1]Note that MathWebSearch does not index subformulae natively, therefore we have to replace each formula φ in D with the set of `dbid`s of subformulae of φ