

Semantic Search for Quantity Expressions

Guided Research Proposal

Tom Wiesing
Supervisor: Michael Kohlhase
Jacobs University, Bremen, Germany

January 28, 2015

Abstract

In this proposal we describe how to approach Semantic search for Quantity Expressions, in particular how to make the existing MathWebSearch system aware of physical units.¹

EdN:1

1 Introduction

In this paper we want to give an approach to Semantic Search for Quantity Expressions. A quantity expression is a number² together with a physical unit, for example $25 \frac{\text{m}}{\text{s}}$ where $\frac{\text{m}}{\text{s}}$ the unit. This quantity expression is equivalent to $90 \frac{\text{km}}{\text{h}}$ (with equivalent in this sense meaning it expresses the same quantity) although the units are different. In a semantic search for quantity expressions we want to be able to search for a certain quantity expression and find any equivalent ones as well.

EdN:2

MathWebSearch (MWS for short) is an existing semantics-aware system to search \LaTeX documents¹ for mathematical formulae [2]. As it is semantics-aware it not only searches for formulae in a simple-minded “text search” way but also includes simple transformation rules, such as $a + b = b + a$. Additionally it can deal with wildcards such as $x + \sqrt{x}$. For this query MWS would deliver formulae as above where x has been substituted with any sub-formular.

¹EdNOTE: Continue and finish abstract

²EdNOTE: Find a better expression for this.

¹Technically, MWS itself can only search XHTML documents. However with the help of \LaTeX Xml [6] it also handles \LaTeX documents.

So far the transformation system has been used by MWS exclusively for mathematical formulae. In this paper we propose an extension for quantity expressions which should prove useful for physicists. The end user will search, for example, 100°C and also get results which show 212°F or 373.15K .

This proposal is organised as follows: In section 2 we shortly describe the existing MathWebSearch system and then proceed in section 3 to describe in detail the proposed extension. Finally in section 4 we discuss challenges as well as related work.

2 An overview of the existing MathWebSearch system

As mentioned above, MathWebSearch is a search engine for formulae. MWS consists of 3 main components as well as a frontend² [4].

The backend consists of 3 main components,

1. a crawler,
2. a core system and
3. a public rest API.

The crawler, as its name suggests, crawls corpora for formulae. For each corpus MWS uses, a separate crawler has to be implemented. The crawled formulae are passed to the core system and indexed in an MWS Harvest. The core system is also responsible for parsing queries and sending results back to the REST API. This is done by searching the harvest only.

Because MWS is semantics-aware, the harvest does not only contain the exact formulae that can be found in the original corpus but also all versions that are equivalent to it. These are generated with the help of MMT and theory graphs.

MMT is a **Module system for Mathematical Theories** [7]. In MMT we can represent mathematical theories, views and imports.

A mathematical theory in this sense is a set of symbols along with a set of axioms³. In their MMT representation both the symbols and axioms have types.

²The frontend is not part of MWS directly but rather built on top of the REST API, more on this later.

³While in mathematics axioms do not require proof, in MMT it is possible for an axiom to actually be a theorem that has a proof.

Theories can be related in two ways, with imports and views. If theory A imports theory B then theory A contains all symbols and axioms from theory B. As an example, the theory of a *commutative groups* imports the theory of *groups*. The *commutative group* theory additionally has the axiom of commutativity. We can consider an import as a type of inheritance relation. In contrast to this a view is a type of instance relation. For example, there is a view from the theory of *Monoids* to the theory of *Natural numbers*. A theorem in one theory can be translated along views as well as along imports. For example, given a theorem in the theory of groups, we automatically get a theorem in the theory of commutative groups.

The theories and their relations can be represented with the help of a graph. This graph is then called theory graph. In the following example, we represent imports as solid edges and views as dashed edges.³

EdN:3

Every theory in MMT has a (globally unique) URL, called the MMT URL. Sometimes it is useful to not write down a theory explicitly, but only give an existing theory and a view to this new theory. We then call this new theory an *induced theory*. The MMT URL of this induced theory is only given via the view. However, the MMT URL contains enough information that MMT can generate all symbols, axioms and induced theorems explicitly [3].

4

EdN:4

The frontend for MathWebSearch, which is not part of MWS itself but running client-side in a web browser, is written in HTML5, CSS and JavaScript. It accesses the REST backend and depends on MathML support to render Mathematics. When the client enters a \LaTeX formula to search for, the \LaTeX ml daemon [1] is used to transform the query in content MathML. Next, the client renders the MathML (to show the formula the user is searching for) and then sends the query off to the MWS API. Upon receiving results, the client renders them and links to the original documents.

There are several implementations of frontends and crawlers as well as extensions of MathWebSearch. One particular implementation is capable of crawling the arXiv corpus, which contains approximately 750.000 documents. A list of demos can be found at [5].

³EDNOTE: Have an example of a small theory graph here with help from @miancu.

⁴EDNOTE: What is a formula in MWS HARVEST? Continue here

3 The goal - a semantic search for quantity expressions based on MathWebSearch

The goal of the guided research is to get a complete system that can perform a semantic search of quantity expressions on a specific corpus. As such it needs 4 components: (1) *A crawler* that searches through documents and finds units which can be indexed, (2) *a core system* that takes care of storing indexes and processing queries and is aware of units with the help of a theory graph, (3) *a rest API* that has to have a way of sending units from and to a client and (4) *a frontend* that runs in the web browser.

Furthermore the system will be flexible with respect to

- *the quantity expressions it understands*, Adding new units should be a simple process so that the system can be easily extended to also capture rare units and
- *the corpus* which should be exchangeable easily.

The crawler will be based on existing MWS crawlers. As it is very difficult to find and mark up all units in a corpus automatically, we will restrict ourself to an artificial corpus containing only lorem ipsum documents with randomly inserted units. This does not solve the problem of writing a crawler, however it will enable us to build a proof-of-concept system first.

The core system will almost completely be inherited from existing implementations. As it does not search formulae, it has to be made aware of unit translations. This will be done with the help of theory graphs. It is not yet clear how exactly these will look, a speculation can be found below in 4.

The rest-based API will have to take the user input from the client and pass the entire quantity expression on to the core system. Once it receives a result, it will have to translate these back into human-readable form and then pass these on to the frontend.

As with the existing system, the frontend will be a web page that incorporates 3 main elements:

1. a search input for a number,
2. a search input for a unit, with the format as described below in 4 and
3. a result page that displays results and links to the found documents.

⁵EdNOTE: Maybe have a final paragraph in this section? Or write more?

4 Challenges and related Work

The biggest challenge with the approach will be to find a standardised representation for quantity expressions, mainly for units⁴. Because we want to translate between these representations however, it is insufficient to just have such a representation. We additionally need to find a standardised way of translating between units need to exist as well.

⁶ ⁷

EdN:6

EdN:7

The units will have to be entered in some fashion in the frontend. While it is trivial to design an interface where a single unit can be entered, it is non-trivial when we want to recognise composite units as well. Furthermore units with (si-)prefixes (such as kilo or mega) can either be recognised separately or as part of the unit. There are several input formats that can be used: AsciiMath, L^AT_EX and MathML, to name only a few. Independent of the input format, the end result (delivered to the backend of the search engine) will have to be represented in one way.

Finally, there is the problem of choosing a suitable corpus and implementing a crawler which will find and mark up quantity expressions. The latter will be taken on by ???⁸ in a separate effort⁹. The task of finding a suitable corpus is postponed for now. In case there is sufficient time a corpus can be marked up manually and crawled afterwards.

EdN:8

EdN:9

¹⁰

EdN:10

References

- [1] Deyan Ginev. The L^AT_EXml daemon: Editable math for the collaborative web.
- [2] Radu Hambasan, Michael Kohlhase, and Corneliu Prodescu. MathWeb-Search at NTCIR-11. In Noriko Kando and Hideo Joho and Kazuaki Kishida, editors, *NTCIR 11 Conference*, pages 114–119, Tokyo, Japan, 2014. NII, Tokyo.

⁴Here, standardised means that all quantity expressions are represented in a standardised, machine-readable fashion, not equivalent quantity expressions being represented in exactly one way.

⁶EdNOTE: Mention the graph that has to be created.

⁷EdNOTE: Mention: Ranges have been implemented in MWS by Radu; Quote needed

⁸EdNOTE: Who is doing unit finding?

⁹EdNOTE: Reference here

¹⁰EdNOTE: Write final paragraph

- [3] Mihnea Iancu, Michael Kohlhase, and Corneliu-Claudiu Prodescu. Representing, archiving, and searching the space of mathematical knowledge. In Hoon Hong and Chee Yap, editors, *Mathematical Software - ICMS 2014 - 4th International Congress*, volume 8592 of *Lecture Notes in Computer Science*, pages 26–30. Springer, 2014.
- [4] Michael Kohlhase and Corneliu Prodescu. Mathwebsearch manual. Web manual, Jacobs University.
- [5] Math WebSearch a semantic search engine. web page at <http://search.mathweb.org>. seen September 2008.
- [6] Bruce Miller. **LaTeX**ML: A \LaTeX to XML converter.
- [7] Florian Rabe and Michael Kohlhase. A scalable module system. *Information & Computation*, 0(230):1–54, 2013.