Semantic Search for Quantity Expressions
Guided Research Proposal

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

January 25, 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 number[2] together with a physical unit, for example $25\frac{\mathrm{m}}{\mathrm{s}}$ where $\frac{\mathrm{m}}{\mathrm{s}}$ the unit. This quantity expression is equivalent to $90\frac{\mathrm{km}}{\mathrm{h}}$ (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 documents [3] [1] 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 forumlae as above where x has been substituted with any sub-formular. EdN:3

[1] EdNote: Continue and finish abstract
[2] EdNote: Find a better expression for this.
[3] EdNote: Do not quote the damon but something general here.
[1] Technically, MWS itself can only search XHTML documents. However with the help of LaTeXml [1] it also handles LaTeX documents.

So far the transformation system has been used by MWS exclusively for mathmatical forumlae. In this paper we propose an extension for quantity expressions which will be useful in physical papers. 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[4]: In section 2 we shortly describe and discuss the existing MathWebSearch system and then proceed in section 3 to describe in detail the proposed extension. Finally in section 4 we discuss possible problems with this approach and related work. EdN:4

2 Background - The existing MathWebSearch system

As mentioned above, MathWebSearch is a search engine for forumlae. MWS consists of 3 components as well as a frontend [2] [3].

The backend consists of 3 main components,

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

The crawler, as its name suggests, crawls corpera for forumlars. These formulars are then transformed into a readable format. Next, the core system build a search index and gets fed queries form the public API. It then processes these queries and send results back to the client via the restful API.

The frontend, 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 dependens on MathML support to render Mathematics. When the client enters a LaTeXformular to search for, the LaTeXml daemon [1] is used to transform this into an XHTML query. Next, the client renders the MathML (to show the formular 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.

An extension to MWS, Thema-Search, [5] allows to search for Math and EdN:5

[4] EdNote: Update this if we change the structure

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

[5] EdNote: Quote Them-search

Text simultaniously, howver this done with the help of a seperate search engine.

For each corpus MWS uses, a seperate crawler has to be implemented. One current crawler exists for the ArXiV system, [6], which has a corpus of approximatly ??? documents [7]. The frontend is deployed at ??? [8] [9]. Disadvantages of the current approach EdN:6 EdN:7 EdN:8 EdN:9

- has to be re-generated each time a document is added

3 The proposed extension

The goal of the guided research is to get a complete system that searches a corpus of documents for units as already indicated above. This system should be nicely accessible to the end user. Furthermore, it should be extendable with respect to

- *the unit system,* Adding new units should be easy and most translations should be deducted by the system automatically.
- *and the searchable corpus.* It should be easy to search a different corpus of documents provided units are properly marked up inside it.

As with the existing system, the frontend should be a web page that works in all modern browsers. It should be mobile friendly. It should communicate directly with the backend via a RESTful API. The frontend should incoperate 4 main elements:

1. a search input for a (real) quantity,
2. a search inout for a unit, described further in section 4,
3. an additional search input for text and
4. a result page that displays results and links to the found documents.

The REST-based backend should have 3 major tasks:

1. translate raw unit input into a standardised form (see section 4 for details),

[6] EdNote: Quote this
[7] EdNote: Get an estimate here
[8] EdNote: Quote the frontend here, maybe mention that this is thema-search
[9] EdNote: Give an overview of the advantages / disadvantages here

2. search for documents using the input from the frontend and

3. make the documents available to the enduser.

The corpus [10] EdN:10

- should consist of a lot of tex documents
- should have marked up units
- ideally, if a single document is added, only the new corpus should have to be re-scanned (procedular generation)
- should be easily exchangable

The unit transition system [11] EdN:11

- should be a graph
- should have few connected components and each of the components should be sparse (i. e. few connections)
- translation are:
  - either a factor towards a single unit
  - or a composition of a factor together with a product or fraction of units [12] EdN:12
  - Perhaps include prefixes somehow?

4 Problems and related Work

There are several problems with this approach. [13] EdN:13

The units have to be entered in some fashion in the search engine. 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 si-prefixes (such as kilo or mini) should also be recognised which is an additional problem. There are a few alternatives to use as input formats: AsciiMath, LaTeXand MathML, to name only 3. Independent of the

[10] EdNote: Write form here onwards
[11] EdNote: Write this part.
[12] EdNote: Figure out more details about this
[13] EdNote: Continue intro pargraph

input format, the end result (delivered to the backend of the search engine) should be MathML containing the unit in some yet-to-be-determined standard form.

The unit equivalences should be in the form of a theory graph [14] The equivalence (translation) itself contains both a translation for the quantity and the unit. The values searched for should be simple real numbers together with the forumlar. However since all numbers must be represented in a finite fashion, translations forumlae can gvie problems. A natural quantity in one unit can be a repeating decimal in a another unit. Also rounding of quantities might depend on the unit used. It is thus preferable not not search for an exact value but instead for a range of values. These ranges should depend on the unit. These ranges have just been implemented in MWS by Radu [15]. [16] [17] EdN:14 EdN:15

Finally, there are the problems of finding a corpus and marking up units in this corpus. The second problem, marking up units in a corpus, will be taken on by ???[18] in a seperate thesis. The first problem will be postponed for now. In case there is still time, a suitably corpus can be marked up manually and plugged into the system. For first testing, I will create a dummy corpus of lorem-ipsum style documents which will contain a random sampling of units. EdN:16 EdN:17 EdN:18

[19] EdN:19

References

[1] Deyan Ginev. The LaTeXml daemon: Editable math for the collaborative web. URL: http://latexml.mathweb.org.

[2] Radu Hambasan, Michael Kohlhase, and Corneliu Prodescu. MathWebSearch at NTCIR-11. In Noriko Kando and Hideo Joho andKazuaki Kishida, editors, *NTCIR 11 Conference*, pages 114–119, Tokyo, Japan, 2014. NII, Tokyo. URL: http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings11/pdf/NTCIR/Math-2/05-NTCIR11-MATH-HambasanR.pdf.

[14] EdNote: Reference this properly.
[15] EdNote: Quote needed; reformulate
[16] EdNote: Write excatly how to use ranges
[17] EdNote: Check if the reference in (2) is clear
[18] EdNote: Who is doing unit finding?
[19] EdNote: Write final paragraph

[3] Michael Kohlhase and Corneliu Prodescu. Mathwebsearch manual. Web manual, Jacobs University. URL: https://svn.mathweb.org/repos/mws/doc/manual/manual.pdf.