Units for MathWebSearch* EdN:1

Guided Research Proposal

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

January 1, 2015

Abstract

In this proposal we describe an approach to introduce Units to MathWebSearch [2] EdN:2

1 Introduction

[3] EdN:3

MathWebSearch (MWS for short[4]) is a system to search (latex) documents for mathematical formulae. Additionally it can also search for text in the documents[5]. However it not only searches for formulae in a simple-minded string way but also includes simple transformation rules, such as $a + b = b + a$. Additionally, is is possible to search with wildcards such as $x + \sqrt{x}$. In this example MWS delivers results of the given form where x is substituted with any sub-formular[6]. EdN:4 EdN:5 EdN:6

MWS has been shown to be very useful for mathematicians[7]. The transformation system it uses can currently be used only for mathematical formulae which limits its applications. In this paper we propose an extension for physical units[8]. Instead of transforming mathematical formulae, the search EdN:7 EdN:8

*EdNote: Preliminary Title
[2]EdNote: Write abstract properly
[3]EdNote: Write an introductary sentence / paragraph?
[4]EdNote: should I really use abbreviations here?
[5]EdNote: Or is this the multi-faceted search that is currently planned? Do we really need this sentence?
[6]EdNote: Re-formulate this and link to an example
[7]EdNote: Quote neeeded
[8]EdNote: Reformulate this?

engine should transfer physical units. 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[9]: 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:9

2 Background - The existing MathWebSearch system

As mentioned above, MathWebSearch is a search engine for mathmatical formulars in documents. It has a corpus of ??? [10] documents and is currently deployed and used by ??? and ??? [11]. EdN:10 EdN:11

The frontend, running client-side in a web browser, is written in HTML5, CSS and JavaScript. It accesses a REST backend and dependens on MathML support to render Mathematics. It simply accessing a REST backend via AJAX. When the client enters a LaTeX forumar to search for, the backend renders MathML which is then sent back to the client. Next, the client renders the MathML (to show the formular the user is searching for) and also sends back the MathML to the server to search for it. Finally the server sends back results to the client which then shows a list of them.

The backend, written in ?? [12], as can be seen from the procedure during a common search, has 2 independent components. The LaTeX to MathML rendering is not part of MathWebSearch but rather uses LaTeXML [13]. The searching is handled by MWS directly. EdN:12 EdN:13

For each corpus a big search index is generated before search [14]. Then during runtime only this index is searched. [15] EdN:14 EdN:15

[16] EdN:16

Advantages of this approach / hwo MWS is received [17] EdN:17

Disadvantages of the current approach

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

[9] EdNote: Update this possible if we change the structure
[10] EdNote: Get an estimate here
[11] EdNote: where?
[12] EdNote: What is the backend written in?
[13] EdNote: Really? Reference needed.
[14] EdNote: Explain this better
[15] EdNote: Explain theory graphs here
[16] EdNote: Explain how MWS uses those here
[17] EdNote: Find points here

3 The proposed extension

- want a complete system
- searches a corpus of documents for units
- which is presentable to the end user
- should be extendable with respect to
  - the corpus. Plugging in a new corpus should be as easy as running a script somehwere.
  - the units. Adding new units should be simple by just adding a conversion to one already known unit.

The frontend

- a web page
- should work in modern browsers, preferably mobile-friendly
- should only be a frontend for a REST backend
- has an input for a unit
- has an input for a value
- maybe have facetet search on top

The backend

- REST based
- based on the existing system
- has to have a format of units
- has to receive text queries
- has to receive exact values or ranges or automatically generated ranges

The corpus

- 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

- 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 [18] EdN:18
  - Perhaps include prefixes somehow?

[19] EdN:19

4 Problems and related Work

The unit input system:

- Entering a single unit and recognising it is simple
- It is not clear how to enter composite units
- the end result delivered to the search engine should either be LaTeX or MathML
- maybe allow different inout methods:
  - The output latex
  - AsciiMath (with autocompletion would be nice)
  - MathML?
- system needs to be aware of full unit names as well as abbreviations

Unit translation

[18] EdNote: Figure out more details about this
[19] EdNote: Write this

- Should just be rational factors
- might give a problem with rounding
- maybe have ranges instead
  - this has just been implemented by Radu [20] EdN:20
- support for composite units: $a \cdot b$ and $\frac{a}{b}$.

Finding a corpus

- We need to have a suitably large corpus of documents to test this properly
- the units need to be marked up in the corpus
- actually finding them is done by ??? [21] EdN:21
- The results should show which unit is originally in the text and also show the value in the unit searched for.

[22] EdN:22

[20] EdNote: Quote needed
[21] EdNote: Who is doing the unit finding?
[22] EdNote: Write this