OMEX: Software for Mining Mathematical Expression Semantics from Scientific Documents

Yiannos A. Stathopoulos
Department of Computer Science
Oxford University
Oxford, UK
yiannos.stathopoulos@mathb0x.com

Brian Harrington
Department of Computer Science
Oxford University
Oxford, UK
brian.harrington@cs.ox.ac.uk

***Abstract*—Semantic analysis of scientific documents can benefit from the information carried by mathematical expressions. However, making established data-mining techniques formula-aware is pre-conditioned on the ability to process expressions in documents. In this work, we present OMEX, a software framework capable of extracting mathematical expressions from scientific documents produced using the LaTeX typesetting environment.**

I. Introduction

The volume of scientific documents available on-line has been extremely beneficial to semantic analysis research. The vast majority of this work, however, is focused on extracting knowledge from textual data. Undoubtedly, text carries a considerable amount of information. However, in the context of scientific documents, mathematical expressions and their structure are also valuable kernels of information. The ability to extract and process semantics of mathematical expressions from documents is not only useful directly (e.g. for creating formula references automatically) but may be the foundation for more intelligent semantic analysis of scientific documents.

The Oxford Mathematical Expression Extraction (OMEX) framework presented in this paper is a software system capable of mining formulae in PDF documents. Given mathematical expressions laid out for print, OMEX extracts their underlying semantic meaning and constructs representations suitable for computer processing. Particular emphasis is given to PDF files generated using the LaTeX environment since it is the preferred tool for generating scientific documents.

II. Motivation and Demonstration

Mathematical expressions carry much of the information to be communicated by scientific literature. In order to exploit this information mathematical expressions must first be extracted from documents. Achieving this initial milestone to facilitate further exploration of information retrieval, semantic analysis and understanding of scientific documents has been the primary motivation behind the construction of the OMEX software system.

The OMEX software is intended to run at the back-end, supporting a server-side service (e.g. a search engine). For the purpose of this demonstration, a command-line version of OMEX will be presented. Visitors will have the opportunity to observe the system collect mathematical expressions from a set of documents and format them in real-time into numbered lists as illustrated in figure 1. Additionally, visitors can learn how the system works using explanations of inter-process output, mostly in the form of images illustrating key decisions. With this demonstration, we intend to highlight the difficulties behind mathematical expression recognition and raise interest for scientific document semantic analysis.

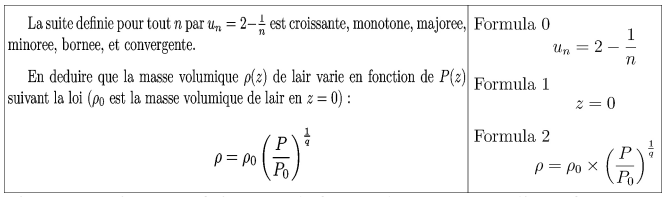

Fig. 1: Snippet of input (left) and corresponding formatted output (right)

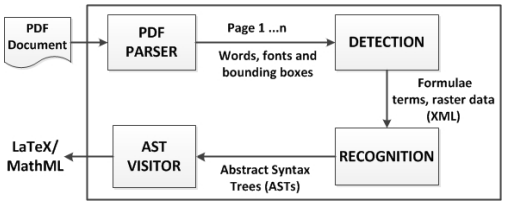

Fig. 2: Overview of OMEX Architecture

III. The OMEX Framework

Processing in the OMEX framework is organized into a pipeline where the output of one process becomes the input of subsequent processes. Each process in this pipeline is required to operate on well-defined input and output. Thus, replacement and/or inclusion of new processes can be done without disrupting the entire pipeline. As a result, the OMEX framework is extensible and encourages experimentation. Implementation has been carried out under the UNIX programming environment using C++.

The pipeline, illustrated in figure 2, is composed of two main processes labeled *detection* and *recognition*. The first process is responsible for identifying formulae and isolating their constituent symbols. Text, font and raster data (bounding boxes) of constituent symbols are accumulated on a page-by-page basis into a portable XML document. This document is passed on to the recognition process for structural analysis. The pipeline produces internal tree structures which can be

"flattened" into the desired output format. This approach allows for fine-grained customizations and decouples the PDF and expression parsers.

A. Detection

Once a page is parsed by the PDF parser, font data, text elements in the page and their positions are passed on to the detection processes. A simple page-flow analysis algorithm is applied to cluster text elements into logical groups, based on proximity. Mathematical expressions in LaTeX-generated PDF documents can appear either *inline* (in the same line as text) or *displayed* (placed on their own line). In order to distinguish mathematical expressions from surrounding text, we employ a *flood algorithm*.

The idea behind the flood algorithm is that a formula can be regarded as a lake, with its sub expressions being connected regions of this lake. Similarly, the text surrounding a formula can be considered as the dry land around the lake. Using this metaphor, we gradually introduce water (iterative propagation of the state) in the lake, which flows into its connected regions. As the regions (sub-expressions) get flooded with water, the lake (formula) boundaries gradually emerge, separating the lake from the land (surrounding text).

The detection process accumulates the elements of each mathematical expression on a page-by-page basis into a well-defined XML file. Each expression is rasterized and identified components are matched up against corresponding bounding boxes in the raster image. Raster data is particularly useful for filling-in missing text and symbols during the recognition process.

B. Recognition

First, the recognizer parses the XML output of the detection processes into internal data-structures. Subsequently, term and raster data for each expression is processed by a chain of modifiers. Modifiers have to be implemented using an abstract interface with well-defined input/output types. As a result, additional modifiers can be implemented and gracefully attached to the chain as the system matures. Currently, the chain supports optical character recognition (OCR), through the Tesseract library [1], for filling-in missing text.

The updated data is passed on to a *sequencer*. At this point, the elements of an expression are organized into an adjacency graph based on positional relationships and distance. Each adjacency graph is transformed into an adjacency "mesh" using a fixed-point computation that iteratively re-writes edges based on a set of pre-defined rules. Next, the mesh is traversed with edges explored according to a specific priority.

The resulting sequence of terms and raster bounding boxes is "consumed" by an expression parser. Parse functions for mathematical structures are built on top of a set of core services. These services form a layer of abstraction from the parser's internals resulting in cleaner implementations of parse functions.

Formulae in printed material are laid out in two dimensions and have semantics that deviate from expressions found in programming languages. For example, $f(x) = \frac{x}{3} + 5$ means $f(x) = (x/3) + 5$ rather than $f(x) = x/(3 + 5)$. Other deviations include omissions of binary operators (implicit application) and unary operators assuming higher precedence than binary operators (e.g. -3^2 means $(-3)^2 = 9$ and not $-(3^2) = -9$). Parsers intended for recognizing printed mathematical expressions must be aware of these deviations in order to extract the intended meaning. Our parser enforces the semantics of printed mathematics by employing geometric constraints while parsing mathematical constructs.

IV. Applications

As a standalone software system, OMEX has a number of direct applications. For example, the system can be used to collect formulae occurrence statistics from scientific literature. In addition, existing mathematical search engines such as MathWebSearch, [2] and ActiveMath [3] can employ our software for unsupervised mathematical expression extraction. Traditional text-based information retrieval (IR) systems, on the other hand, are based on a bag-of-words (unordered list of keywords) approach and cannot take advantage of the information carried by the structure of formulae. Using our technology, such IR systems can be augmented with mathematical knowledge, enabling scientists to express their information need accurately and obtain more relevant results.

Combined with natural language processing (NLP) techniques, mathematical knowledge can be used in document analysis and clustering systems for disambiguating the domain of scientific documents and overloaded terms in science. For example, whether the term "field" refers to the algebraic notion or to the Physics definition of the term would be made clear from the mathematical context. Successful integration of mathematical knowledge into existing systems may be the first step in developing software for "mathematical understanding", capable of identifying, disambiguating and associating mathematical terms and definitions.

V. Conclusion

OMEX is an experimental software framework that implements unsupervised data-mining of mathematical expression semantics from scientific documents. Due to abstractions in its core services and fine-grained customisability, OMEX is extensible and adaptable. As a result, it can form the basis for more advanced analysis of mathematical expression semantics. The final goal of our demonstration is to convey our enthusiasm about the technology and its potential to the community.

References

[1] R. Smith, "An overview of the tesseract ocr engine," in *Proceedings of the Ninth International Conference on Document Analysis and Recognition - Volume 02*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 629–633.

[2] M. Kohlhase and I. A. Sucan, "A search engine for mathematical formulae," in *Proc. of Artificial Intelligence and Symbolic Computation, number 4120 in LNAI*. Springer, 2006, pp. 241–253.

[3] P. Libbrecht and E. Melis, "Methods for access and retrieval of mathematical content in activemath," in *Proceedings of the ICMS-2006*, N. Takayama and A. Iglesias, Eds., no. 4151. Springer-Verlag, 2006.