EuDML

The EUROPEAN DIGITAL MATHEMATICS LIBRARY

ICTPSP

ICT POLICY SUPPORT PROGRAMME

part of the Competitiveness and Innovation Framework Programme CIP

DELIVERABLE

Project Acronym: EuDML

Grant Agreement number: 250503

Project Title: The European Digital Mathematics Library

D7.1: State of the Art of Augmenting Metadata Techniques and Technology

Revision: 1.2 as of 2nd November 2010

Authors:

Petr Sojka	MU, Masaryk University, Brno
Josef Baker	UB, University of Birmingham
Alan Sexton	UB, University of Birmingham
Volker Sorge	UB, University of Birmingham

Contributors:

Michael Jost	FIZ, Karlsruhe
Aleksander Nowiński	ICM, Warsaw
Peter Stanchev	IMI-BAS, Sofia
Thierry Bouche, Claude Goutorbe	CMD/UJF, Grenoble
Nuno Freie, Hugo Manguinhas	IST, Lisbon
Łukasz Bolikowski	ICM, Warsaw

Project co-funded by the European Comission within the ICT Policy Support Programme		
Dissemination Level		
P	Public	✓
C	Confidential, only for members of the consortium and the Commission Services	

Revision History

Revision	Date	Author	Organisation	Description
0.1	31th October 2010	Petr Sojka	MU	First prerelease.
0.2	2nd November 2010	Josef Baker	UB	OCR stuff added.
0.3	11th November 2010	PS,JB,AS,VS	MU, UB	Restructured.
0.4	23th November 2010	Petr Sojka	MU	TB's part added.
0.5	27th November 2010	Petr Sojka	MU	Near to final polishing, TODO removal, release for language and stylistic checking.
1.0	27th November 2010	Petr Sojka	MU	Release for internal review, with language checked by AS and NER part by ICM.
1.1	28th November 2010	Alan Sexton	UB	Minor corrections.
1.2	2nd December 2010	Petr Sojka	MU	Release for EU with comments from JR, MBa, and with part added by HM and extended conversions part.

Statement of originality:

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

Contents

Executive Summary . . . 3

1 Introduction **3**
1.1 Preliminaries . . . 3
1.2 Metadata . . . 4
1.3 Enhancements . . . 4
1.4 Toolset . . . 5
1.5 Implementation . . . 5
1.6 Overview . . . 6

2 Retro-Digitized Data **6**
2.1 Image Capture . . . 6
2.1.1 Bitmap Storage Formats . . . 8
2.2 OCR . . . 8
2.2.1 Tesseract . . . 9
2.2.2 FineReader . . . 9
2.2.3 OCRopus . . . 9
2.3 Specialist OCR for Historical Documents . . . 10
2.4 Mathematical OCR . . . 10
2.4.1 Infty . . . 10
2.4.2 PDF Extractor . . . 10

3 Retro-Born-Digital Data **11**
3.1 Postscript Extraction . . . 11
3.2 PDF Extraction . . . 12

4 Analysis **13**
4.1 Information Extraction – Named Entity Recognition . . . 13
4.2 Mathematical Formula Recognition . . . 14
4.2.1 Segmentation . . . 16
4.3 Formula Parsing . . . 17
4.3.1 Projection Profile Cutting . . . 17
4.3.2 Virtual Link Networks . . . 18
4.3.3 Graph Rewriting . . . 20
4.3.4 Baseline Parsing with Grammars . . . 20

5 Born-Digital Data **22**
5.1 Data Format Conversions & Editing . . . 22
5.1.1 Conversion of Mathematics from TEX/LATEX Sources . . . 22
5.1.2 Tralics . . . 22
5.1.3 TEX4ht . . . 23
5.2 Translating PDF to Presentation Formats . . . 23

6 Metadata and Document Refinements: Enhancers **24**

6.1 Metadata Enhancements using Zentralblatt MATH 24
6.1.1 General Methodology 24
6.1.2 Implementation 25
6.1.3 Limitations 25
6.1.4 Applicability to EuDML 26
6.2 Metadata Augmenting tools in NUMDAM and CEDRAM 26
6.3 Metadata Refinement in YADDA Framework 27
6.3.1 ICM YADDA DeskLight Metadata Editor 27
6.3.2 Editing Process of DML-PL 27
6.3.3 Machine Driven Metadata Enriching in ICM 28
6.4 Document Enhancements 28
6.4.1 PDF Optimization by PdfJbIm and pdfsizepot.py 28
6.4.2 Digital Signatures of PDF 28

7 Other Metadata Enhancements by Partner's or Commercial Systems 29
7.1 MU Experience from Building DML-CZ 29
7.2 HDML Metadata Integration 30
7.2.1 Data Cleaning 30
7.2.2 Data Integration 30
7.2.3 Data Provision to EuDML 30
7.3 IMI-BAS's Cyrillic OCR 31
7.4 Commercial Services and Tools 31
7.4.1 Portico 31
7.4.2 Datapage 31

8 Summary, Conclusions 32

Index 37

List of Figures

1 Metadata processing 4
2 A schematic view of the semantic network needed for disambiguation . . 15
3 An example of PPC on $a = \frac{b}{c}$ 18
4 Partial Parse tree for Anderson's Coordinate Grammar [70] 21

Executive Summary
We have identified main issues and challenges on augmenting metadata techniques and technologies appropriate for using on a corpora of mathematical scientific documents. For most partial tasks tools were identified that are able to cover basic functionalities that are expected to be needed by a digital library of EuDML type, as in other projects like PubMed Central or Portico. Generic standard techniques for metadata enhancement and normalization are applicable there.

Deliverable also reviews and identifies expertize and tools from some project partners (MU, CMD, ICM, FIZ, IU, and IMI-BAS). Main (unresolved) challenges posed are *OCR of mathematics* and reliable and robust converting between different math formats (TEX and MathML) to normalize in one primary metadata format (*NLM Archiving DTD Suite*) to allow services like *math indexing and search*.

In a follow up deliverable D7.2 [58], tools and techniques will be chosen for usage in the *EuDML core engine* (combining YADDA and REPOX), or as a (loosely coupled) set of enhancement tools in a linked data fashion.

"A library is the best possible imitation, by human beings, of a divine mind, where the whole universe is viewed and understood at the same time. A person able to store in his or her mind the information provided by a great library would emulate in some way the mind of God. In other words, we have invented libraries because we know that we do not have divine powers, but we try to do our best to imitate them."
Umberto Eco, [23]

1 Introduction

A subject based library like EuDML, to be viewed and understood, needs as rich and accurate metadata as possible, so that its provided content gets best understanding, exploatibility and visibility [3, 37]. The purpose of work package seven is to provide a toolset that will allow enhancements of the article metadata acquired from content providers.

As opposed to publisher's backfiles (e.g. SpringerLink), repositories or regional digital libraries, EuDML is conceived as a *virtual* library over already existing repositories. Tools to be provided are thus not meant for primary metadata editing, but rather for *automatic* enhancements based on the collected critical mass of content, or as triggers for setting up policies for acquiring data from data providers. In this setup, subject based procedures may take place, taking advantage of crawled and collected content, to allow building of global services as similarity matching. In a sense, the whole EuDML might be viewed as one huge metadata enhancer of mathematical content and existing databases such as Mathematical Reviews and Zbl.

1.1 Preliminaries

In parallel deliverables, related notions and terminology is described. In Deliverable D3.1 [14] metadata are classified into subbasic, basic and advanced, and an overview of input/primary metadata from content providers is presented. The EuDML metadata schema is described [34], to be used as the main container for metadata, especially for the EuDML service architecture.

Workpackages 6 to 10 could be viewed in a broader sense as metadata enhancements, e.g. Deliverable D8.1 [38] overviews state of the art in areas like citation linking and matching, document similarity and clustering and document categorisation.

1.2 Metadata

For every document item in EuDML repository there will be pointer to the electronic file of *the* published version, presumably a PDF file. Linked to this *document* there will be metadata, i.e. data about this document. We take rather a broad view on metadata, as *any data related* to the document in hand. They could be primary sources of a document (TEX files or NLM tagged [41] XML files), they may be extracted from a document (plain OCR texts, plain texts with math as TEX, etc.), they may even relate the document to other documents (e.g., list of similar articles).

One can obtain the metadata by various means, as seen on Figure 1. The handling and tools to cope with the [meta]data are quite different, depending on its primary data origin.

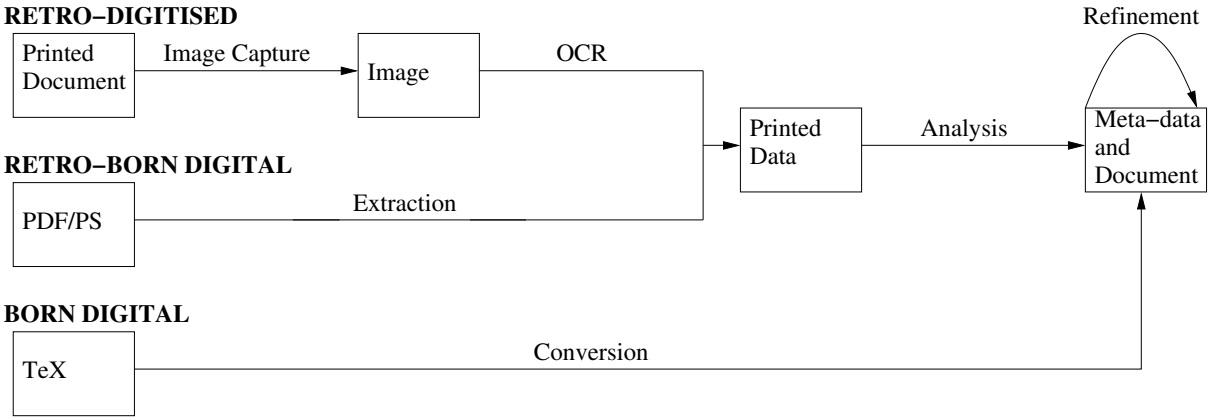

Figure 1: Metadata processing

The general workflow of a typical digitisation project reflects different types of acquired input data:

full digitisation from print: work starts from a paper copy;

full digitisation from bitmap image: work starts from an electronic bitmap of pages;

retro-born-digital: work starts from an electronic version of the document (usually in PostScript or PDF);

born-digital: workflow of the journal production is enriched with a conversion step, [semi-]automated export of journal data for the digital library or archiving purposes.

Content providers thus will deliver primary data in these heterogeneous formats and EuDML enhancers have to cope with this variety and make efforts to unify them into a common format [41].

1.3 Enhancements

By *metadata enhancements* we mean any process that takes [a part of] metadata, and creates a new, possibly enhanced, version, or converts it into another format. All enhanced

versions, uniquely identified, are linked to the EuDML document they relate to. Metadata enhancement proceses can be pipelined, to form a more complex e*nhancement and* u*nification/conversion* tools, *eutools* for short.

Within the project, several eutools will definitely be needed (implementing arrows from Figure 1:

1. conversion tools from TEX to NLM XML (MathML)
2. PDF/PS extractor
3. OCR tool, including math OCR, layout analysis
4. pipeline for scanned image handling

Some eutools may refine metadata chunks or documents:

1. bibliographic segmenter that tags bibliographic items in plain text
2. Zbl lookup that takes a reference string and provides checked metadata from Zbl database,
3. named entity recognizers that identify and tag entities in plaintext or NLM,
4. author disambiguation/clustering,
5. tools for classification of mathematical documents and measuring their similarity [47],
6. PDF optimizer recompressing image objects in PDF with the new JBIG2 compression,
7. batch metadata editor for making a well-defined and safe generic corrections to metadata

or can be offered to data providers to bring their metadata from subbasic to basic level, as

1. web-based metadata editing application,
2. metadata editor, for manual correction of metadata [7],
3. workflow for born-digital publication production with direct export of metadata for DML [49],
4. PDF stamper for digital signing of produced PDF.

1.4 Toolset

Thematic sets of eutools will be collected into *toolsets* [58, 39] according to the planned workflow and EuDML working plan. For eutools related to workpackages 8 to 10 we refer to the respective deliverables, in this document we concentrate on the state of the art of technologies related to the WP7.

1.5 Implementation

The main implementation platforms are YADDA and REPOX systems, that are capable of implementing core services of EuDML funtional specification [10], where eutools implement internal *services* that enhance metadata, or only processing nodes of complex internal enhancing service.

Some eutools may evolve into external services on their own, to be offered to data providers (such as the PDF enhancement service).

1.6 Overview

The structure of this document follows closely the outline defined in Figure 1. Section 2 discusses acquisition of retro-digitized data via image capturing and OCR techniques. We thereby focus on techniques that are used for arbitrary documents and only discuss specialist techniques pertaining to the mathematical or historical nature of documents as far as necessary. Section 3 shall be devoted to the discussion of retro-born digital data focusing on the extraction of processable character information from formats such as PostScript or PDF.

For both retro-digitized and retro-born digital data, actual metadata as well as mathematical expressions have to be explicitly extracted by way of explicit analysis. An overview of these techniques is presented in Section 4. On the other hand, for born-digital documents one generally has full access to the source material, thus instead of analysing re-captured data, the relevant metadata only needs to be located in the source and translated into a common format. This is primarily achieved via specialist conversion tools, which we discuss in Section 5.

At this point all metadata that can be explicitly extracted from documents is available in a similar format, if not of similar quality. As a consequence it has to be subsequently refined and enhanced. A summary of the experiences with techniques for metadata extraction and enhancement by the partners in the EuDML project is presented in Section 7 before concluding the document in Section 8.

2 Retro-Digitized Data

Retro-digitised data refers to documents for which only paper or bitmap images scanned from paper, is available. If only paper versions are available, then these documents must first be scanned to a bitmap image. Various processes can then be applied to the images in order to improve the accuracy of further analysis. This can include noise reduction, de-skewing, de-warping and binarization. We refer to the original scanning plus such image refinement processes as *image capture*. Image capture is the responsibility of the data provider and is not part of the EuDML project.

Following image capture, the image must be analysed to extract and identify characters, symbols and graphic elements. This process we refer to as *optical character recognition (OCR)*.

2.1 Image Capture

Images are usually captured using scanners and stored as TIFF images, either bi-tonal (1-bit depth) or grayscale. Best practice manuals agree that bi-tonal images should have at least 600 DPI. In the case of grayscale (4 or 8-bit depth) images, 400 DPI may be sufficient.

Operations performed on these images are:

1. *geometrical correction* such as narrowing the baselines and widths of the same characters on the same line;
2. *cropping* the page to cut out the speckles at page borders;

3. *Wiener* or *Sigma* filter, to reduce background noise;
4. *blur filter*, 3×3 pixels, to eliminate one or two pixel size variations;
5. *binarisation* with manually adjusted parameters for each batch (usually journal volume);
6. *despeckle filter*, with both white and black spotting, 3×3 pixels;
7. *publish/export*: processed TIFFs are stored being compressed by the Lempel-Ziv-Welsh or Zip (Deflate) method for compressing grayscale and the G4 one for binarized images to speed up further processing (OCR) and to save space.

Both the order of these steps and the parameter adjustments for images of different quality are very important. Slightly different operations are needed if the input files are already bi-tonal and some filters are applicable only on grayscale images.

Step 1 employs the algorithms that allow perspective correction of a scanned image. As most of the material to digitize cannot be cut, 2-up page spreads are scanned, making the text size non-uniform even when trying to flatten the spread by pane of glass. Some tools such as Book Restorer can also flatten the lighting across the scanned spead. For more details of this step see [15, page 2].

Step 2 crops the unnecessary border parts of the page shot.

Step 3 reduces background noise from the image.

Step 4 aims at better binarisation and despeckling by unsharping the shapes in the image.

Step 5 is necessary as most OCR engines work on bi-tonal images. It may be left to the high-quality OCR engine—clever thresholding starts to be a standard part of OCR programs [54], or perform it ourselves adaptively based on OCR feedback [46].

Step 6 is inserted to remove small impurities in the image.

Step 7 is the final step: image is stored as LZW-compressed grayscale or G4-compressed bi-tonal TIFF.

For lower resolution data slightly different operations are needed as the input files are already bi-tonal (e.g. if upscaling is carried out before unsharping) and because some filters are applicable only on grayscale images.

It is wise to differentiate processing of pages with grayscale images (e.g. photos) so that they are not degraded by image filters suitable for text. To avoid possible difficulties in the later steps it is important, from the very beginning, to carefully check image quality before proceeding with the remaining steps. At least automated procedures that check the technical metadata (e.g. `tiffinfo`) and image quality (pixel width and height) has to be the part of quality assurance. Metrics of compressibility by the JBIG2 encoder can be used to trigger quality checks.

There is a tradeoff between price of image cleanup and quality results within the constraints of a digitisation budget. When acquiring a craftmanship in good image editing software, results very close to (or even better than) the original can be achieved [53]. These hand made touches are usually beyond the budget of most digitisation projects, where as the highest degree of automation is needed to reduce the cost of digitisation.

In DML-CZ [22], Book Restorer™ image restoration software by i2S was used for interactive and batch image processing in an uncompressed TIFF format. Sets of batches of typical transformation procedures to be used by Book Restorer™ operators to achieve the best price/effort ratio were prepared [44].

Fine-tuning of operations on the pixel level pays back in the following step: the OCR.

2.1.1 Bitmap Storage Formats

Most legacy image software uses TIFF format, either LZW-compressed, ZIP-compressed G4-compressed or uncompressed to speed up processing. Tools tend to move to PDF workflows, allowing state-of-the art compression methods as JPEG2000 or JBIG2, taking advantage of the speed of today's processors.

The greatest part of PDF documents size is taken by images (if there are any). Since PDF version 1.4, there is support in PDF [1] for standard JBIG2 which significantly improves compression ratios for bi-tonal images. As bitmaps stay to be used in the final delivery PDFs (with a possible *below-image* text layer used for indexing and search), they can take significant amounts of bandwidth when downloaded from digital libraries. For this reason, it *is worth optimizing* bitmap storage and compression methods.

JBIG2 is a standard for compression of bi-tonal images developed by the Joint Bi-level Image Experts Group. Such images consist only of two colors (usually black and white). The main domain of such images is scanned text. JBIG2 was published in 2000 as international standard ITU T.88 [66], and one year later as ISO/IEC 14492 [17].

It typically generates 3–5 times smaller files than Fax Group 4 and 2–4 times smaller than JBIG1 (previous standard released by Joint Bi-level Image Experts Group). JBIG2 also supports lossy compression that increases compression ratio a few times without noticeable visual differences in comparison with lossless mode. Lossy compression without noticeable lossiness is called *perceptually lossless* coding. Scanned text mostly contains flyspecks (small dirts) and, by getting rid of them, quality of output image can be even better.

2.2 OCR

Optical Character Recognition is a technique used to identify glyphs within scanned documents or images, and to produce the appropriate electronically encoded characters, usually in ASCII or in Unicode [63]. This is often combined with other techniques to produce further analysis, such as identifying words, formulae, tables, diagrams and other structural elements within the document.

The most highly researched area of OCR is the recognition of plain text written in a Latin script, and there are many high quality commercial and open source software applications available, which when used in conjunction with high quality documents, can achieve extremely high recognition rates. In this case, a high quality document would have a number of attributes, namely

- Well printed, on a good paper, with glyphs made of solid unbroken lines.
- Well typeset, in a common font with no overlapping of characters, consistent spacing.

- Well scanned, at a resolution of around 600 DPI or above, with minimal noise and skew.

However, documents frequently lack some of these attributes and many users require more than just plain text recognition, for example, the identification and analysis of elements such as tables, formulae and metadata. These tasks require more advanced techniques and are the focus of much current research.

There are several OCR tools available, either free (*OCRopus*, *Tesseract*, *Ocrad*, *GOCR* or *CuneiForm*) or commercial (*FineReader*, *Omnipage*, *AnyDoc Software*, *Brainware*, *ExperVision*, *Readiris*, *ReadSoft*, *SmartScore*, *Infty* or *Simple OCR*).[1]

2.2.1 Tesseract

Tesseract [24] is a free software optical character recognition engine that was originally developed as proprietary software at Hewlett-Packard between 1985 and 1995. After ten years without any development taking place, Hewlett Packard and University of Nevada, Las Vegas released it as an open source in 2005. Tesseract is currently in under development by Google and released under the Apache License, Version 2.0. Tesseract is considered one of the most accurate free software OCR engines currently available.

2.2.2 FineReader

FineReader OCR Engines are commercial products by ABBYY. FineReader recognizes 195 OCR languages, including: 37 main languages with Latin, Cyrillic, Greek or Armenian characters, East Asian languages and 113 ICR languages.[2] FineReader Engine's recognition API delivers special features for pattern training and creation of user languages. It does not and probably never will support math OCR, though.

2.2.3 OCRopus

OCRopus is currently being developed under the lead of Thomas Breuel from the German Research Centre for Artificial Intelligence in Kaiserslautern, Germany and is sponsored by Google. OCRopus is developed for Linux; however, users have reported success with OCRopus on Mac OS X and an application called *TakOCR* has been developed that installs OCRopus on Mac OS X and provides a simple droplet interface.

OCRopus is an OCR system that combines pluggable layout analysis, pluggable character recognition, and pluggable language modeling. It aims primarily for high-volume document conversion, namely for Google Book Search, but also for desktop and office use or for vision impaired people.

OCRopus used to use Tesseract as its only character recognition plugin, but switched to its own engine in the 0.4 release. This is especially useful in expanding functionality to include additional languages and writing systems. OCRopus also contains disabled code for a handwriting recognition engine which may be repaired in the future.

1. See `http://en.wikipedia.org/wiki/List_of_optical_character_recognition_software` for overview and links.
2. For full list of recognition languages see `http://finereader.abbyy.com/recognition_languages`

OCRopus's layout analysis plugin does image preprocessing and layout analysis: it chops up the scanned document and passes the sections to a character recognition plugin for line-by-line or character-by-character recognition.

As of the alpha release, OCRopus uses the language modeling code from another Google-supported project, *OpenFST* [48], optional as of version pre-0.4.

2.3 Specialist OCR for Historical Documents

The EU IMPACT project [19] focuses specifically on OCR of historical documents, as state-of-the-art OCR engines have to be adapted in order to cope with the challenge of historical fonts and layouts. No commercial or other OCR engine is able to cope satisfactorily with the materials published since Gutenberg by the start of the industrial production of books in the middle of the 19th century [19].

Historical documents contain specific archaic fonts and notation, and usually scanning is carried out in higher precision to not only get information content but also page image details for study by historians.

2.4 Mathematical OCR

Standard OCR software generally performs very badly with mathematical texts, Fateman et al. [28] found that recognition rates sometimes dropped to only 10% when presented with well typeset, two dimensional equations. Heuristics for text in straight lines do not transfer well into two dimensional structures, coupled with the much larger character set which contains many similar characters are the two main reasons for this.

This has led to a number of math specific OCR engines being created, including [27, 62, 16], along with a large database of mathematical characters which can be used in conjunction with mathematical OCR [52]. The results of the OCR are usually used together with formula parsing tools, which can help to choose the correct candidate for each character. This can be very computationally expensive, with recognition speeds far slower than for standard text.

2.4.1 Infty

In experimentation based upon 26 scientific articles, Infty [62] achieved character recognition rates of 99.44% for ordinary text and 95.81% for mathematical text. For the structural analysis of mathematical expressions, 89.6% were perfectly analysed. However, when used in conjunction with perfect character information, it achieved a rate of 97.9%.

2.4.2 PDF Extractor

Baker et al. [5] tested their system on 128 mathematical expressions taken from two freely available mathematical texts. 3 expressions produced semantically incorrect results when parsed. 18 produced results which rendered slightly differently to the original, but with no loss of semantic information. The remainder where all parsed correctly, with no semantic or rendering differences.

3 Retro-Born-Digital Data

Retro-born-digital data comprises documents that were originally created electronically and the corresponding generated format (usually PDF or PostScript) are available, but for which the original sources that produced the electronic format are lost or otherwise not available. Consequently the content of such articles is not easily accessible and available for further content processing. In order to extract and enhance the metadata for these documents they have to be restored into a suitable format. Examples of retro-born-digital documents are articles for which the PostScript or PDF format is available, but no longer the LaTeX or Word source from which they were produced.

There are a large number of tools available that can convert content from PostScript or PDF files into regular text. From simple open source tools such as Unix command line programmes like `ps2txt` or `pdftotext` that produce simple ASCII output, to online tools that translate PDF files to Word documents or to commercial products like ABBYY's PDF transformer that converts PDF to various MS office formats. While these tools are useful to extract content from straightforward text documents with varying degree of reliability, none is suitable to work with mathematical documents as is necessary in the EuDML project. Therefore our discussions below will concentrate on tools specific for extracting mathematical data from PostScript and PDF files.

There have been two main approaches to extracting information directly from electronically born documents for the purpose of mathematical OCR, the first by Yang and Fateman [68] which deals with PostScript files, and the second by Baker et al. [4, 5], which works with PDF files.

3.1 Postscript Extraction

Yang and Fateman [68] present work that enables processing PostScript documents and converting the contained mathematical expressions into Lisp expressions. The processing of the PostScript document was based on a modified version of a PostScript to text converter, which adapted the GhostScript interpreter to output information about words and their bounding boxes. They were particularly concerned with the operators that painted glyphs and lines, changed fonts and translated or scaled the coordinates. When a document had been passed through the interpreter, a series of instructions were output, which were followed to produce a string of characters, along with their font name and bounding box. Whilst this was generally a trivial task, certain characters had to be identified through pattern matching groups of commands. This occurred when glyphs were constructed of multiple commands, common with square roots, integral signs and division lines.

As a result the PostScript extractor translates mathematical expressions into corresponding Lisp expressions, which are suitable for further processing. In fact the Lisp expressions can be understood to constitute parse trees in their own right, which could be the basis for translation into other formats. However, there is no evidence that any meaningful further processing or translation has been attempted by the authors.

Currently the extractor is no longer readily available for download and use. Moreover, it is outdated in that its implementation relyed heavily on a Lisp engine with some C++ routines. Its reliance on specific pattern matching techniques make its

accuracy dependent on the actual mathematical subject area of documents and thus unsuitable for general purpose extraction in a digital mathematics library. More importantly however, while there are still documents available in Postscript format in many digital libraries, Postscript itself is falling into disuse as an archival format. Moreover, modern software makes conversion to PDF not only easy but also lossless in terms of the structure of the content. As a consequence the use and further development of extraction techniques based on the PDF format should be preferred in the context of our project.

3.2 PDF Extraction

The approach by Baker et al [5, 6] works with PDF files, in particular those making use of Type 1 fonts. The tool is able to extract the name, size, font name and style, baseline coordinates and bounding box of each character within the file. In order to achieve this, image processing of a rasterised version of the file in conjunction with analysis of the PDF is carried out.

Initially a clip is made which discovers the bounding boxes of every single glyph. The dimensions of the clip along with its page number and associated file are passed to the PDF extractor. The extractor uses this information to find the correct page within the file in order to start the extraction process.

Each page within a PDF file has a content stream, which holds instructions for placing and displaying characters, lines and images, along with a number of other resources, including font dictionaries, which contain, directly or indirectly the name and family of a font, the names of all of its characters and various other information. In the first pass over the file, all of this information is collated.

In the second pass, the content stream is processed. Each time a text environment is entered within a content stream, the font and font size are both stated, followed by a command that transforms the current position and the byte value for each character to be displayed. When the procedure encounters byte values, the appropriate character is obtained from the font extracted in the first pass. The remaining instructions within the text environment are then sequentially processed. In addition to text environments, content streams also contain drawing environments, these are also processed as the lines drawn here often also represent characters, such as fraction lines and other extendible characters.

At this stage, even though the exact information on the characters within the PDF file has been obtained, the exact bounding boxes of the characters is not known. Whilst the baseline information is sufficient for recognising simple text on a given line, more accurate data is required when trying to calculate 2D relationships between characters, which is necessary for formula recognition. This task is completed by registering the characters with the bounding boxes obtained during image analysis. This involves overlaying the connected components and character information, implementing special rules to deal with symbols consisting of more than one connected component, such as i, $=$, and symbols consist of more than one PDF character, which tends to occur with large extendible characters such as roots and brackets and large operators.

4 Analysis

From the identified characters, symbols and graphic elements obtained by image capture and OCR from retro-digitised data or from extraction from retro-born digital data, words, sentences, mathematical formulae and other textual and mathematical content can be discovered. The role of the different content elements, i.e. as titles, authors, citations, displayed or embedded mathematical expressions etc., must then be extracted. We refer to this process by the generic name of *analysis*. In general, analysis works by accumulating low level elements, i.e. character or symbols, into higher level ones, i.e. words, formulae etc., using some variation of a parsing or recognition process. Many different approaches exist and differences between them often reflect the varying nature of the type and quality of low level elements that they work on. For example, if the low level character identification arose from an OCR program applied to a retro-digitised bitmap image, then the analysis must account for uncertainty in the identification of those characters. This might be done by including approximate dictionary lookup techniques to correct character identification in the context of the words that they appear in. If, on the other hand, the character identification arose from extracting the precise information from retro-born digital PDF documents, then such techniques may not be necessary.

In [5], the information extracted from the available mathematical content from PDF files is used to produce the formula in linearized form employing a grammatical approach. The grammar is based on an extended and updated version of a grammar defined by Anderson for recognition of handwritten mathematics [2]. This results in an output string that is available for further processing which can be carried out by specialist drivers.

4.1 Information Extraction – Named Entity Recognition

A particular scenario of information extraction is the recognition of entity references (such as authors, headings, citations, persons, organizations, places, time periods, etc.) and their resolution to real world entities. In this scenario, the entity references are found within unstructured text or in semi-structured data fields. The typical output of this scenario is the exact location of where the entities are mentioned within the data, and their resolution to a set of entities known to be disambiguated – text is enriched with appropriate tags or attributes.

Previous research on information extraction has focused mainly on natural language processing. Information extraction processes are composed of subtasks, each task simplifies the text by transforming it into more machine processable structures. So, for example, before entities can be recognized it may be necessary to identify the language of the text, to tokenize the text into paragraphs, sentences and words, and to classify the words for their part-of-speech category. It is by reasoning on the output of this process that references to entities are identified. This process is therefore dependent on evidence given by the natural language text to identify a reference to an entity, and also its type.

Information extraction relies mainly on natural language processing. It consists of a process composed of subtasks that simplify the text by transforming it into more machine processable structures. Some of the typical information extraction subtasks are:

Entity recognition: recognition of entity names (persons, organizations, places, temporal expressions, numerical expressions, etc.).

Coreference resolution: detection of coreference and anaphoric links between text entities.

Word sense disambiguation: identifying which sense of a word (i.e. meaning) is used in a sentence, when the word has multiple meanings.

Terminology extraction: finding the relevant terms for a given corpus.

Relationship extraction: identification of relations between entities (such as person X works for organization Y).

Entity recognition is one of the major subtasks of information extraction.

Because of the complexity of the problem, entity recognition is typically studied on restricted domains. The design of an entity resolution system needs to take into considerations the characteristics of the corpus where it is to be applied, such as the languages, textual genres, domains and types of entities.

In general terms, any *entity recognition* approach consists of a two step process. It starts with the identification of pieces of information which are relevant for solving the entity recognition problem, and then follows by reasoning on the available information to make the decision of recognizing an entity and its type. These pieces of information are typically referred to as *features*, *clues* [51] or *evidences* [18], in the research literature.

In the context of scientific papers we may need to recognize the following entities:

Document: a document in the collection.

Contributor: an author of a given document.

Affiliation: an affiliation in a given document.

Reference: a reference to another document in a given document.

Person: a concrete person (groups multiple contributions).

Institution: a concrete institution (groups multiple affiliations).

Tagged entities like Contributor, Affiliation and Reference may help to *disambiguate* Authors, Documents referred in in-text References etc., and new entities can be identified and distinguished, as seen in Figure 2. Based on the context, a Contributor may be disambiguated into a Person, Affiliation may be identified into Institution, or Reference may be disambiguated into unique Document. "A word is known by the company it keeps." (*Noscitur a sociis*) For more on this topic, see Deliverable D8.1 [38, section 2.2].

4.2 Mathematical Formula Recognition

Many mathematical and scientific documents are available in a digital form; these may consist of scanned journals and papers, word processed and typeset files, PDF files and web pages. With the growth of the internet it is now far easier to publish such material, thus the number of documents is increasing at a very large rate. Many of these documents are disorganised and, especially when published in PDF format, are poorly indexed with limited search facilities available. Thus the mathematical knowledge contained in these documents is difficult to retrieve, since it is neither searchable nor easily accessible. Indeed mathematics search is still in its infancy, and, due to the lack of index-able material, its scope is severely limited. However, with the availability of much more indexed material, robust mathematical search engines could be developed. Other advantages of having fully

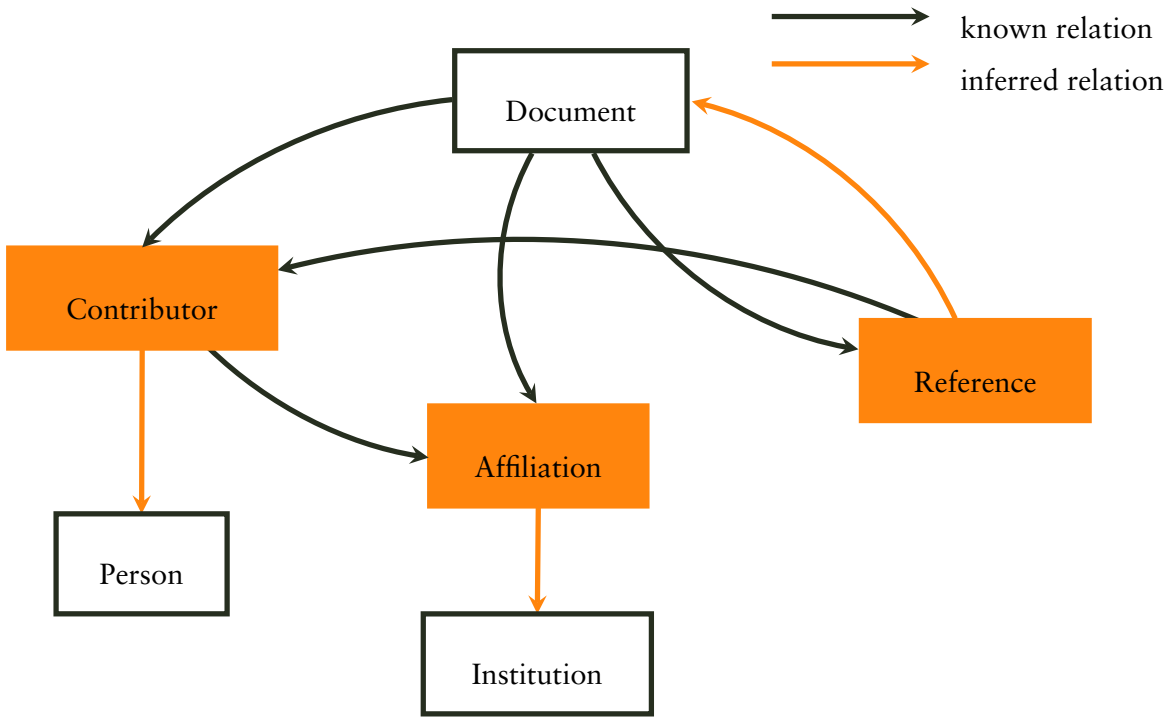

Figure 2: A schematic view of the semantic network needed for disambiguation

accessible mathematics within documents is the ability to interact with other software, such as screen readers and equation editors.

For mathematical notation to be made fully accessible, it needs to be written in a way that can be understood by a computer, i.e. following a strict syntax, which describes both the context and its representation. Even though suitable languages such as *MathML*, *OpenMath* and *OMDoc* exist, they are rarely used, especially in electronically published papers and journals. Instead, the mathematics is represented by a mixture of images—which is common in HTML, or by the use of symbols from many different fonts and character sets in an unusual format; this is common with PDF, Word and PostScript. These problems are often highlighted if one tries to copy and paste a formula into a new document. Instead of the formula being accurately copied, the result is usually a series of nonsensical symbols, or, more often, nothing at all.

Whilst much research and work has been completed on formula recognition, many problems still exist which mainly stem from the fact that it is very difficult to recognise mathematical symbols precisely. This is because there is a far greater number of symbols used in mathematics compared to Latin alphabet languages, they are often set in many different fonts and styles and many of the symbols look very similar. Coupled with the lack of a lexicon, which could be used to help correct recognition errors, this leads to recognition rates far lower than when processing text. These errors are then propagated through the formula recognition process, making it very difficult to correctly parse and understand a formula.

There are three distinct phases in formula recognition. The first is segmentation, in which any mathematics is marked as such, in order to be processed separately from any text, diagrams or other parts of the document. The second stage is the recognition of the constituent symbols of each formula. The output from this stage depends on the user requirements, but will comprise at least of the character name and may also include its font name, weight, size, style and baseline. The third and final stage is the parsing of these symbols in order to produce either a parse tree of, or a marked up version of the formula, this could be in any number of formats such as LaTeX, MathML, OpenMath etc.

4.2.1 Segmentation

In order to try and recognise the mathematics, it first has to be segmented from the rest of the text. This task is not trivial as formulae may be typeset inline, embedded in a line of plain text, or it may be separated from the plain text in a displayed expression. This introduces a degree of ambiguity over what should, or should not, be marked as mathematics.

Manual Segmentation Probably the most simple, but time consuming method of identifying mathematics within a document is to manually segment formulae from the surrounding text. Human error will play a factor in this technique, and users have to be trained and follow strict guidelines in order to consistently and accurately segment the mathematics.

Baker et al. [5] make use of this method, having an application which allows a user to select areas of mathematics upon the page for further analysis. Formulae with interspersed text are clipped in their entirety, and preceding labels and trailing periods can also be clipped where they exist. Their parsing system allows for this text to be included.

Heuristic Based Segmentation Fateman [26] uses a number of heuristics in order to segment the mathematics. Three passes are made over a document, each time moving symbols between a *math bag* and a *text bag*.

In the initial pass, all bold and italic text, mathematical symbols, numbers, brackets, dots and commas are put into the math bag, with everything else being classified as text, including unrecognised characters.

The second pass aims to correct for too much math. The symbols are grouped together based on proximity, then any unmatched parenthesis, leading and trailing dots and commas and isolated 1s and 0s are moved to the text bag. The rule for isolated 1s and 0s is to compensate for recognition errors.

The third and final pass aims to correct for too much text. Essentially, isolated text surrounded by mathematics and mathematical keywords such as sin, tan are moved from the text bag into the math bag.

In later work, Yang and Fateman [68], extracted character information directly from PostScript files rather than using OCR. This gave them access to font information, which they used to add to and improve the heuristics used in each pass. In particular noting that documents created using LaTeX used Computer Modern Math and Computer Modern Roman to typeset mathematics.

OCR Based Segmentation INFTY [61, 62] is a system for full document analysis and thus process text as well as mathematics. The document is initially passed through commercial OCR software, which in conjunction with a large lexicon produces good recognition results of standard text. However it often fails when presented with the various fonts, varying baselines and unusual character found within mathematics, and will often produces meaningless strings with a low confidence. This is used as the initial stage in segmentation.

The second stage involves overlaying results of the OCR onto the original document. The bounding box position and size of each recognised character is then compared with the original. If they vary above a certain threshold, then these characters are also treated as mathematics. This method helps to identify when there are changes in baseline, so can identify when expressions containing sub and superscripts have been misrecognised as text.

4.3 Formula Parsing

4.3.1 Projection Profile Cutting

Projection Profile Cutting [42, 67] (PPC) is a method used to obtain the structural layout of a mathematical formula, by recursively separating components of a formula via horizontal and vertical projections in order to construct a parse tree. It is traditionally applied as a preprocessing step before OCR on a scanned image, but can also be used after OCR.

Given a mathematical formula, PPC first performs a vertical projection of the pixels in the formula onto the x axis, in order to find white space that horizontally separates the components. The white space indicates the position where the formula can be cut vertically into components that are horizontally adjacent. Each of the discovered components is then, in turn, projected horizontally onto the y axis in order to separate its sub-components vertically. This procedure is repeated recursively until no further cuts can be performed. The remaining components are then considered to be atomic. However, this does not necessarily mean that they are only composed of single glyphs.

The result of the PPC is a parse tree that represents the horizontal and vertical relationship between the atomic components. That is, the first layer, given by the vertical cuts, represents parts of the formula that are horizontally adjacent; the second layer, computed via horizontal cuts, represents the components vertically adjacent, etc.

As an example we observe PPC for the simple formula

$$a = \frac{b}{c}$$

which is given in Figure 3. The first projection leads to vertical cuts that separate the expression into the three components a, $=$, and $\frac{b}{c}$. This corresponds to the first layer in the resulting parse tree below. Here a is already an atomic component; thus it can not be cut any further and becomes a leaf in the parse tree. $=$ and $\frac{b}{c}$ on the other hand can be further cut horizontally, as given at the top of Figure 3. This leads to a subsequent level in the parse tree, by introducing a new branch for each vertical component. E.g., cutting $\frac{b}{c}$ yields three branches, one for each glyph in the expression.

While in our example the parse tree is very small, PPC can easily scale to more complex expressions. Indeed PPC is a fast simple way to effectively perform more complex layout analysis of mathematical expressions [70]. However, it is has some significant drawbacks:

1. As can observed with our example, PPC separates multi glyph characters, which is generally undesirable, as they have to be reassembled.
2. PPC may not identify super and sub scripts, e.g. $a^2 + 4$ may have the same parse tree as $a2 + 4$, because the expression is reduced into its atomic components with just five vertical cuts. Therefore sub and superscripts require additional processing.
3. Another problem occurs when dealing with enclosed characters, the most common example of this happens with square roots. In the case of $\sqrt{a} = 2$, neither a horizontal nor vertical cut can separate the $\surd$ and the a. Thus $\sqrt{a}$ is viewed as an atomic components and wrongly becomes a tree node in the parse tree.
4. Poor quality images also cause significant problems, skew can alter horizontal and vertical relationships, touching characters can be recognised as atomic components and broken lines can lead to too many nodes.

Raja et al. [45] adapted PPC, using it as a post rather than preprocessing step to OCR, thus reassembling the structure of an expression. By using it after the completion of OCR, knowledge of the glyphs can be used to overcome the problems of enclosed characters and multi glyph characters. It also allows cuts to be quickly computed, rather than searched for within the raw image data, which significantly improves performance.

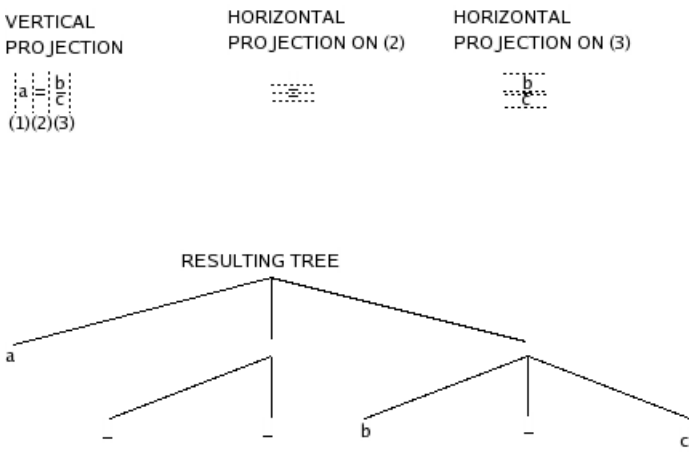

Figure 3: An example of PPC on $a = \frac{b}{c}$

4.3.2 Virtual Link Networks

Suzuki et al. [25, 62] use a virtual link network for formula recognition. This works by constructing a network of characters represented by vertices linked together by a number of directed, labelled edges with costs. Once this is complete, the tree with the lowest cost is returned. The technique has the added advantage of being able to correct recognition errors incorporated by OCR software.

Initially, OCR and layout structure analysis is completed, which is used to identify the characters and their relative positions. Thereafter, the normalised size and centre of each character is calculated. This is used to determine possible spatial relationships between characters, such as superscript, subscript, those on the same line etc.

When this step is complete a Virtual Link Network is created. This is a network where each node represents at least one possible character, and each link shows the parent child relationship between candidates for each node.

The OCR software used can return up to 10 different choices of character per node, each with a likeness value between 0, lowest match, and 100, highest. The top 5 matching characters will be used as a candidate for each node, providing its match value is over 50.

The different relationships that can exist between nodes are:

- Child is next character on the same line as Parent;
- Child is right or left super or sub script of Parent;
- Child is in upper or under area of Parent (Fraction);
- Child is within a root symbol of Parent;
- Child is under accent symbol of Parent.

Costs are associated with each link, which increase the more that they disagree with the results of the initial structural analysis step. E.g., if the structural analysis step determined that two nodes were horizontal, a horizontal relationship would have a lower cost than a sub or super script relationship, also the cost of a link from a character with a low match value would be higher than a character with a high match value

Admissible spanning trees are then searched for, and outputted if they have a total cost below a predetermined level. An admissible spanning tree has to meet the following four criteria:

1. Each node has a maximum of one child with the same label;
2. Each node has a unique candidate chosen by the edges linked to it;
3. The super or subscript sub tree to the right of a node K; are left of the horizontally adjacent child of K;
4. The super or subscript sub tree to the left of a node K; are right of the horizontally adjacent parent of K.

Once the list of admissible candidate trees is created, their costs are reevaluated, adding penalties if certain conditions are met. These conditions are generally based around unusual relationships between nodes. Once this final step has been completed the tree with the lowest cost is returned.

Whilst they conclude that it is a robust technique which works with various printing styles, and is very good at correcting errors when characters have different normalised sizes, it does have problems when different characters have the same normalised size, e.g., Ss, O0o and 1l would be grouped together, whereas SS and ll would be distinguished.

If this technique were to be combined with data extracted from the PDF, then the erroneous grouping together of characters would no longer be an issue. This is because the information derived from the PDF file indicates whether a letter is capitalised or not. Also, the whole process would be significantly faster. As a node would be given a single candidate character from the PDF analysis, rather than a choice from OCR, there would be far fewer spanning trees that would be created and need to be evaluated.

4.3.3 Graph Rewriting

Graph grammars have been widely used for diagram recognition in many different areas, including music notation, machine drawings and mathematical notation [29, 36]. The general idea is to create a graph of connected nodes, and then use graph rewriting rules to replace sub graphs, much like a standard grammar.

Lavirotte and Pottier's graph grammar technique uses information returned from the OCR software, such as bounding boxes, sizes, names and baselines of the characters within a formula, to build a graph. The graph consists of vertices and edges.

A vertex V consists of: a lexical type, such as operator, variable, digit, etc., its value and a unique identifier. A graph is defined as a finite set of connected edges containing at least one edge. This ensures that all vertices appear in at least one edge.

To build a graph, first, an attempt is made to link each symbol to another in 8 directions, top, bottom, left, right, top-left, bottom-left, top-right and bottom-right. Secondly illegal edges are removed, this is achieved by following rules for each type of symbol, e.g. top-left and bottom-left incoming edges are not allowed with Σ. Finally a concept of gravity is introduced, which determines the strength of links between symbols. For example, the 1 and + in 1+ would have a strong force between them but a weaker force in 1^{+}.

Graph grammars are then used to parse the graph. A graph grammar is specified by a start graph and a set of production rules. The aim of the grammar is to replace sub graphs with other sub graphs, until a single node remains, which contains the syntax tree of the formula. Context sensitive grammars are used, this helps to prevent ambiguities that may exist when more than one rule is available.

A bottom up parser is used in this technique, for two main reasons, one is to enable it to formulae to be treated separately (when more than one occurs upon a sheet), which reduces complexity. The second reason is that it is very difficult to discover the parameters, such as the bounding boxes of components of a formula, in a top down manner.

Lavirotte notes that the grammars used are non trivial and that heuristics are sometimes required to remove ambiguities, also that the system will need to be adapted to incorporate error detection and correction, as in real applications, OCR is unable to recognise characters with 100% success. Zanibbi [70] comments that graph grammars are very computationally expensive and that the approach taken by Lavirotte and Potier restricts the expressiveness of the grammar.

4.3.4 Baseline Parsing with Grammars

One of the most common techniques for parsing mathematics involve variations of string and coordinate grammars, along with modified techniques for parsing, based upon baselines within the notation.

The baseline is usually the centre line through a character, or string of characters, though it sometimes refers to the writing line or a thicker band through the characters. Also, it may be called the mainline or mid-line depending on the techniques and terminology used.

Anderson [2] produced the earliest work in this area, which he called a coordinate grammar. This is similar to an attributed string grammar. The input consisted of an

attributed symbol list with bounding box coordinates and centre line coordinates. Figure 4 shows the rule for a fraction, which is matched when a horizontal line is found with symbols both above and below it, that do not extend past the line horizontally, or overlap it vertically.

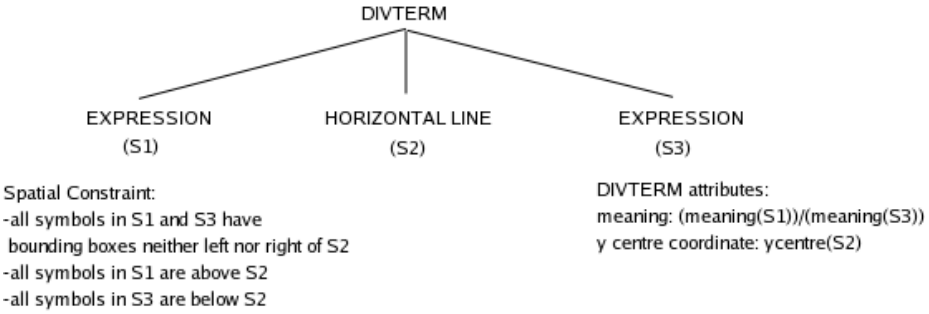

Figure 4: Partial Parse tree for Anderson's Coordinate Grammar [70]

When a valid parse tree is found, the semantics of the notation are obtained by propagating the meaning of the leaves to the the root of the tree. In the case of Figure 4, coordinates would be assigned to the whole term and the meaning would be calculated from S1 and S3.

Early research into these grammars, including Anderson's, tended to be used solely to analyse formulae [27, 28, 43]. However more recently they have been combined with other techniques for verification and error correction [64, 65, 71]. The reason for this is that structural analysis purely based on grammars will fail if any character or symbol recognition errors are included [65]. Andersons grammar worked, but only under the assumption that there were no recognition errors contained within the input.

This means that the grammars are often applied to a formula that has already been analysed, then converted into another format, such as LaTeX. With the knowledge that the expressions to be analysed are limited to those produced in a certain way, it means that a guaranteed parse tree will exist, and that variance in notation can be ignored [27]. The results of the grammar can then be used to deduce a semantic meaning of the formula, or to inform the user that recognition errors have occurred.

Both top down and bottom up parsers can and have been used. Anderson used a top down parser, which was very computationally expensive, and Fateman used a bottom up parser which was much faster—though it worked on a smaller subset of mathematics. Anderson's grammar always assumed perfect input, and Fateman's was only able to deal with noise free input. Fatemans proposed that a top down approach would be far better to deal with more realistic, noisier input.

As Toyota [65] declared that grammars would always fail when presented with erroneous input, it means that they tend to be used for verification of other techniques. However, using perfect character information extracted directly from PDF files, Baker et al. [6] have been able to overcome this problem. Using this data together with an extended and heavily modified version of Anderson's Linearize algorithm, they can convert two dimensional mathematical formula into a linear string representation. A standard Yacc-

style LALR parser is then used to analyse the resulting expression and produce an abstract syntax tree.

5 Born-Digital Data

5.1 Data Format Conversions & Editing

There is a dual world of formats for mathematics: in the authoring world, the easily extensible namespace of LaTeX macros and shortcuts is used. In the world of software applications, MathML is a standard that allows interapplication communication between math-capable programs like symbolic algebra systems or rendering engines of web browsers.

The main problem with conversion utilities for mathematics is that the namespace used in the authoring world of LaTeX is not fixed, and has to be restricted to some subset supported by the conversion application.

NLM format allows storing of dual representations of the same entity, e.g. both LaTeX and MathML description of math formulae. Given the available format (usually author's LaTeX) we will need to convert it to another (MathML for indexing), or even OpenMath for disambiguation applications.

5.1.1 Conversion of Mathematics from TeX/LaTeX Sources

There are many converters of this sort available, with varying purposes and quality. There are standalone systems, limited to small subsets of LaTeX, such as *Gellmu*, *HTMX*, *HTeX*, *Hevea*, *HyperLaTeX*, *HyperTeX*, *LaTeX2HTML*, *LaTeX2hyp*, *Ltoh*, *Ltx2x*, *Math2HTML*, *TechExplorer*, *TeX2HTML*, *TeX2RTF*, *Texi2HTML*, *Tth*, *Vulcanize*, and *WebEq* or tools seeding DVI, and extracting text with markup, such as Elsevier's *LaTeX2SGML*, and Gurari's *TeX4ht*, or replacing the TeX engine in some way, e.g. MicroPress *TeXpider*, *Omega* by Haralambous and Plaice, and Grimm's *Tralics*.

Tools from the first category, e.g. *LaTeX2HTML* support only some standard LaTeX markup, without supporting macros. This, inevitably, sooner or later breaks, as authors' creativity in macroprogramming, naming and introducing shortcuts makes conversion unmanageable, and a full TeX macroexpansion engine is needed for robust conversion.

Full TeX expansion is employed in the case of the *Tralics* and *TeX4ht* converters, which we describe in more detail.

5.1.2 Tralics

Tralics is a LaTeX to XML converter. In the context of EuDML, it will be used to translate mathematical formulas extracted from a PDF file and encoded in LaTeX to their MathML equivalent.

Tralics is free software, see mainly `www-sop.inria.fr/miaou/tralics/`

Tralics is currently used by some partners of the EuDML project (UJF/CMD, DML-CZ, Zentralblatt MATH).

The main work in EuDML will be to adapt Tralics to our particular needs (through configuration files) and to define an API that will integrate the software into the set of

tools provided by WP7, to be used e.g. in the EuDML user interface so that queries could contain mathematics in LaTeX notation.

5.1.3 TeX4ht

Eitan Gurari's TeX4ht [30] works by using TeX proper as a conversion tool, storing the input markup in hidden parts of the typeset output, from which it is then converted. It is used in the publishing workflow of River Valley Technology, Inc. to convert 200,000 pages of STM journal content into validated XML with MathML per year. From XML, LaTeX is generated again to get the final PDF or *ePub* format for electronic delivery of ebooks.

5.2 Translating PDF to Presentation Formats

Baker et al. [5, 6] further manipulate the linearized form of mathematics expressions extracted from PDF files to produce a variety of output formats. This is essentially a two step process: First parse trees are generated from the linearized string which are used as intermediate representations for subsequent translation into mathematical markup. Different types of parse trees can be generated, from simple parse trees that only hold the basic information on the relationship of symbols in a formula, to more complex parse trees which incorporate also information on font, character sizes, and relative spacing between symbols.

The primary drivers for translation into mathematical markup implemented by Baker et al concentrate on the reproduction of formulas in LaTeX. There exist essentially two drivers: One driver produces LaTeX code that as faithfully as possible reproduces the original formula taking spatial information etc. into account. A second driver aims more at generating LaTeX that closely resembles code that could have been written by a human mathematician. While the latter does not necessarily reproduce exactly the same formula as in the original document, it has the advantage that its LaTeX lends itself more to a semantic evaluation as well as, in some cases, cleaning up potential layout mistakes introduced by the author.

Further drivers consist of a modules producing Presentation MathML, as well as a driver that generates input for Festival, a speech synthesis tool. Most drivers currently focuses primarily upon the reconstruction of mathematics for presentation. While there have been some attempts at semantic interpretation of the parsed mathematical content [6], such as translation into Content MathML, those drivers are not yet fully developed and implemented. Nevertheless, the PDF extraction tool has the advantage that its formula recognition is general purpose in that it is independent of a particular mathematical domain. While the employed grammar might not be all encompassing, it is easy to extend with additional rules. Furthermore the extraction tool is readily available. It relies on a well supported, open source Java tool for decompression of PDF files and on postprocessing routines implemented open source in Ocaml.

The ultimate goal for the next decades is to have PDF to NLM conversion, inverting the usual publishing workflow, e.g. typesetting from fine-grained logical markup into various presentation formats.

6 Metadata and Document Refinements: Enhancers

Metadata collected from various sources and by various means have to be stored, together with their provenance and origin. A *Metadata Registry* (MDR) is a central location (e.g. repository) in an organization where metadata is stored and maintained in a controlled environment. The "registry" part implies that metadata is subject of registration within the Registry, by specifying the set of rules, operations, and procedures that apply to an MDR. It provides a unified view, promotes a common understanding of the information managed in an organization and assists organizations in the sharing and exchanging of mutually agreed information. As a result an MDR can promote harmonization, standardization, use, and reuse of the information throughout and between organizations.

The notion of metadata in MDRs relates to the information needed to describe *information models* (abstract description of how information is represented in a specific context, e.g. a business area, an organization or a community) and data models (representation of an information model for a specific syntax and/or storage method). One particular kind of information that can be registered within an MDR is the rules modelling the relationships and equivalences between data models. This information is commonly called *mappings* and is particularly important for cross-walking between data models.

In the context of EuDML, the MDR can be used for managing and publishing the range of data models and terms used by the various institutions that will provide content. The primary purpose of the *EuMDR* is to improve metadata interoperability by allowing:

- The documentation of the EuDML public and internal data models;
- The documentation of the data models used by the EuDML data providers;
- Transformations among the various data models.

The documentation of data models is done in Deliverable [34]. In the rest of this section we describe and analyze transformations not covered in previous sections, that may bring new metadata quality, with exception of those covered in Work package 8 [39] (enhancers of document *sets*) or 10 [60] (enhancers of document access).

6.1 Metadata Enhancements using Zentralblatt MATH

One of the defined goals of Work Package 7 of the EuDML project is "to validate and merge the discovered metadata with that already registered for the target items."

As a very large repository of metadata for mathematical publications, Zentralblatt MATH can server as a source of metadata that are valuable to digitization initiatives when assigning suitable metadata to scans of the original literature.

This section is reports on the state of the art and best practises that have been developed at Zentralblatt MATH during earlier projects, and how these may be applied in EuDML.

6.1.1 General Methodology

Zentralblatt MATH has successfully applied methods of metadata enhancements for several digital mathematics library projects and digitisation initiatives such as RusDML, and working with a variety of partners such as the Library of Lower Saxony and the University Library of Göttingen and the German National Library of Science and Technology (TIB Hannover).

The usual mode of operation is the following: Scanning and digitization initiatives carry out the scanning of the printed material and, during this process, attach just a small subset of metadata to the scans, e.g. means of journal identification (journal name, ISSN), volume/issue numbers, and page ranges. Further information, such as author, title, classification, keywords, may be available from Zentralblatt MATH, and automatically be identified and carried over to the scans, thus saving capacities that would otherwise be needed for manually re-typing such metadata. Also, direct links to Zentralblatt MATH reviews of the papers can be attached to the scans.

On the other side, the digitization initiative reports back to Zentralblatt MATH about the availability of the scanned full text on the net, so that Zentralblatt can use this information to provide direct links to the full text in the Zentralblatt MATH database. Also, errors inside the Zentralblatt MATH data, and information missing or otherwise unavailable on the side of Zentralblatt (such as translations/transliterations), are reported back.

6.1.2 Implementation

Two main models of operation have been applied so far and can serve as models for EuDML:

1. the digitization project requests a defined subset of Zentralblatt MATH data based on journal name and time period. This subset of Zentralblatt MATH data corresponds to a certain scan run of the digitization project, and should be matched 1:1 to the scanned articles.
2. the digitization project uses online interfaces of the Zentralblatt MATH database to obtain relevant information on the fly. Generic HTTP query interfaces are in place that accept suitably encoded query information such as journal identifier, volume/issue number and page ranges, and return XML-encoded Zentralblatt MATH database items.

6.1.3 Limitations

Zentralblatt MATH covers the worldwide mathematical literature since 1868. Some (currently about 600) mathematical journals receive cover-to-cover treatment, i.e. every article from every journal issue is registered in the Zentralblatt MATH database. The above outlined methodology may be best applied to this set of mathematical journals, since one can be sure to find metadata for every such journal article. There are, however, other journals from which only those articles are selected that have specific mathematical contents. As an example we mention the Arab Gulf Journal of Scientific Research, from which only the (small) fraction of papers explicitly dealing with mathematics are selected for inclusion in Zentralblatt MATH. We expect, however, that the majority of journals handled by EuDML will be in the category of cover-to-cover reviewed journals in Zentralblatt MATH.

Another limitation stems from the business model that governs Zentralblatt MATH: Zentralblatt is ready to provide academic initiatives with the metadata needed for their retrodigitization efforts, in particular bibliographic information (author, title, source) and contents descriptions (mathematical subject classification and keywords). However,

Zentralblatt does not agree that other information, especially reviews and summaries, be copied from Zentralblatt and re-used by the digitization initiative.

6.1.4 Applicability to EuDML

The methodology outlined above has been successfully used for various projects and can be re-used without much ado in EuDML. Since EuDML project characteristics and EuDML partner needs should be taken into consideration, some final clarifications and adaptations will be neccessary for the two methods:

1. agree on actual delivery formats, especially the respective item identifiers;
2. agree on HTTP interface specifications and implement necessary changes in the Zentralblatt MATH online service.

Finally, clear understanding and agreement must be reached about which Zentrablatt MATH metadata are re-usable by the digitization initiative, and which not.

6.2 Metadata Augmenting tools in NUMDAM and CEDRAM

CMD (which is the union of UJF/CMD and CNRS/CMD) have developed a number of augmenting tools and strategies while running the *NUMDAM*, *CEDRAM*, *mini-DML* and *Gallica-MATH* projects, especially for retro-born digital and born digital items [13].

For articles published in the CEDRAM platform, although all of them are prepared in some flavour of LaTeX by their publishers, it was not always possible to get fine grain metadata (e.g. author's first and last names), thus a new input structure based on amsart.cls was created, that permits automatic generation of the metadata from a journal issue compilation (aka *CEDRICS* system). Thanks to the use (and customisation) of Tralics for generating XML structures, all character encodings are coherently converted to Unicode, and mathematics is stored in a dual format: HTMLized LaTeX and MathML [11, 12]

The accumulated know-how, especially regarding the use of Tralics for generating LaTeX and XML from a variety of TeX sources will be contributed to the project.

Combined with an XSLT stylesheet, this system produces EuDML ready metadata in NLM format from a large array of TeX input formats: it produces element-citations from structured bibliographies (in *bibTeX*, amsref,...) and mixed-citations from LaTeX "thebibliography" environment; it can also be used in "retro-born digital mode", where it won't typeset the article's body, but will only be used to regenerate well-structured metadata. A number of use cases for this emerged in CEDRAM as well as in NUMDAM where some TeX source was available to produce more accurate metadata than from OCR or PDF extraction. In some instances, OCR or PDF extraction was used to create a CEDRICS driver file, structured according to ad hoc heuristics, that would be then edited manually to fix mistakes. In other cases, a TeX formula was extracted, processed with Tralics, and stored back as MathML in the XML file (this would use the alternative element in NLM DTD).

Native, retro-born, and XML enhancer modes have each been used on large corpora of documents (few thousands of each flavour).

Matching as an Enhancement Device

1. For good matching, part of it is breaking a given reference string into probable fields–this could be used in itself to help generate basic metadata for items that lack them, such as Gallica's article.
2. Matching an item and pulling context or more metadata from the match might help improve item's metadata.

Enhancements for matching are in more detail covered in an overview D8.1 [38].

6.3 Metadata Refinement in YADDA Framework

6.3.1 ICM YADDA DeskLight Metadata Editor

As a part of the YADDA system suite, the *DeskLight* application has been developed. DeskLight is an metadata editor focused on editing metadata in *BWMeta format*. DeskLight is a stand-alone Java desktop application for database editors and redactors. It may be used also to provide access for journal redactors to store metadata directly in the YADDA repository.

DeskLight may operate as an editor for remote or built-in (embedded) repositories. This allows support for different editing processes, depending on custom database preparation workflows. If used in off-line mode, data is imported into the editor, and the user edits database on his/her own computer. When certain portions of metadata is ready, then the data is exported and imported to the central repository. In online operation, data is edited directly in the repository and result is visible immediately on the web server.

For online operation there is sophisticated security provided, to support different user roles and access rights. Access control allows restricting users to some subset of the database (for example only one journal). It is also possible to enforce verification of data edited by less experienced users by database redactors.

DeskLight provides a number of custom editors for certain metadata fields. BWMeta is rich data format, where each field has a complex structure. DeskLight makes editing this type of record relatively easy, in a modern, user friendly way. It also provides full support for keyword dictionaries, classifications (standard and user generated), some support for authors catalogue and many other features. DeskLight also provides graphical control over certain aspects of the remote YADDA repository. It provides graphical user manager, process control and alternative importing facility for YADDA.

DeskLight is used as an editing tool for a certain amount of scientific bibliographic databases in ICM. It was used to prepare the books collection of DML-PL.

6.3.2 Editing Process of DML-PL

DML-PL has aggregated some mathematical knowledge from Polish scientific journals over the years. The first version was created using a proprietary solution based on an SQL database. This involved input of around 95% of the current journal material. This data has been imported directly to the YADDA system.

DML-PL books have been added to the YADDA system directly, using the *DeskLight* metadata editor. Also new corrections are performed with DeskLight.

A certain amount of the metadata has been obtained from TEX articles' source files. It was produced using bespoke scripts, but it generates top-quality metadata [69].

6.3.3 Machine Driven Metadata Enriching in ICM

ICM does not work on metadata enriching based on OCR or direct extraction from documents. This is due to the fact that most of the data served and processed by YADDA is provided by journal publishers (Elsevier, Springer etc.), and metadata is of top quality.

Because the amount of data provided by the publishers is huge, there is ongoing work on automatic metadata enchancement with citation resolution. This work includes parsing citations and recognizing its format, and matching parsed citations in the database.

Currently we use a regular-expression based engine for citation parsing, which supports all standard and a large amount of non-standard citation formats. Its quality is measured as over 90% accuracy. Also there is ongoing work on providing a citations resolution engine based on machine learning systems. This work currently has very promising results (up to 98%).

The matching step success rate depends on database scope and time extent, and also it seems to provide very good results.

This engine is used operationally on databases served by ICM on YADDA platform.

6.4 Document Enhancements

Primary document optimization and enhancement may also be possible as not all data providers may have tools and expertize to bring documents in optimal shape. The most often used format PDF e.g. allows various compression method to be used for bitmap image, PDF may contain font glyphs that were not used and are needed for document rendering,.... As example of one such a document enhancer may serve PDF re-compressor and optimizer.

6.4.1 PDF Optimization by PdfJbIm and pdfsizepot.py

A PDF re-compressor tool called *pdfJbim* has been implemented taking advantage of extremely high compression ratio of bi-tonal images using visually lossless JBIG2 compression [31, 59]. It was developed using an *improved jbig2enc* encoder. *Jbig2enc* [35] is an open-source encoder written by Adam Langley with the support of Google. It uses the *leptonica* library [9] for image manipulation. Leptonica takes care of rendering text, comparison of symbol components,... Jbig2enc supports only arithmetic coding and, instead of halftone region, uses generic region.

Together with other programs, namely *pdfsizeopt.py* by Péter Szabó, it is reported that, on the data of DML-CZ [22], one can decrease PDF storage size and transmission needs by 62% by a combination of both programs. The tool reduces the size of original already compressed PDFs to 48% of original size.

6.4.2 Digital Signatures of PDF

Program *pdfsign*i [31] makes it possible to sign the PDF using a certificate, in batch mode. This ensures a recipient of the PDF to verify that it originated from the publisher. This is a standard Public Key Infrastructure (PKI) approach used for digital signatures.

Another types of data enhancements in a digital mathematics library are described in [50].

7 Other Metadata Enhancements by Partner's or Commercial Systems

7.1 MU Experience from Building DML-CZ

MU has gained experience from designing and building DML-CZ [55, 8, 22]. A workflow covering data handling for all three types (retro-digital, retro-born-digital and born-digital) was implemented and several tools were designed and employed during DML-CZ development.

The OCR step is followed by further text processing, and its results are used for editing of metadata and references. Most text processing and metadata enhancements in DML-CZ are centered in the web based application called Metadata Editor.

The Metadata Editor (ME) [7, 40, 21] has gradually developed into a fully-fledged and efficient web application, `https://editor.dml.cz`, that allows simultaneous remote editing according to assigned structured access rights. It supports two levels of actions. On the first one the operator editing the data is provided with page thumbnails so that he can visually check the completeness, scan the quality and configuration of the articles, easily shuffle the pages and cut or merge articles if necessary. On the other level the operator can check the automatically imported metadata, edit and complete them. An integral part of the ME is the module for administration of authority files with authors' names. It enables the most suitable version of the name for the DML-CZ to be selected and to match it with all its other versions.

These functionalities in combination with remote access enable distributing the work among several people on different levels of expertise. GUI allows hired operators (mostly students of mathematics) intuitive work on the entry level. They inspect and correct the structure of complex objects (journal – volumes – issues – articles). Afterwards, they make the initial inspection of the metadata, add the titles in the original languages, provide notes signaling possible problems. Experienced mathematicians then add the necessary translations, complete the missing MSC codes, provide links between related papers. They also accomplish the final revision and validation of the metadata.

We consider bibliographical references as important metadata of every paper. Their availability makes it possible to use professional systems like CrossRef® for cross-publisher citation linking. The work starts from OCR of the text, in which a block of references is found. Citations are tagged by a script based on regular expressions written for the citation style of every journal. The operator then checks, edits and approves the list of paper citations.

For fixing errors that can be safely detected (such as a Mathematics Subject Classification (MSC) code string that is invalid in the MSC 2010 standard) procedures are formulated and coded in XSchema generated also from a web-based interface (forms). Other sets of constraint checkers run as overnight jobs together with updates of the database and metadata statistics and logs useful for the management of Metadata Editor's workflow.

Finally, various detection procedures for possible errors have been suggested, evaluated and implemented for finding anomalous and suspicious content of metadata fields, with lists of warnings generated, including hyperlinks for easy checking by an operator. An important control concerns the integrity of TEX sequences in metadata to assure

seamless typesetting of article cover pages in the later stages: all metadata to be typeset are exported in one big file with unique references to the article, and typeset by XeLaTeX to check the TeX control sequences used in the metadata fields. This ensures that all of the TeX encoded mathematics converts into MathML format smoothly. Similar procedures allow for an efficient and economical increase of metadata completeness and quality.

7.2 HDML Metadata Integration

HDML (Hellenic Digital Mathematics Library) contains content from a number of different prominent sources of mathematical documents. This content is provided by the respective owners using a non automated procedure that entails mainly the dissemination of the actual data and the equivalent metadata. The data are mostly retrospective scanned images of scholarly mathematical documents while the metadata differs from provider to provider. In some cases the use of RDBMs is used while in other cases metadata are in the form of accompanying XML files.

In all cases, these data are required to appear under the unifying HDML service. To that end, research in the Ionian University pertaining to the unification of the aforementioned data has been oriented towards (a) data cleaning, (b) data integration as well as infrastructure provision in order to supply the clean and integrated data to the EuDML project.

7.2.1 Data Cleaning

HDML content providers have databases of mathematical documents optimized to their respective requirements that may include provider specific information and, in some cases, lack general information. Accordingly, these metadata require a "cleaning" process in order to remove content that is of no use to HDML. In addition, in some cases of fragmented multiple metadata for a given provider, cross-reference techniques have been implemented in order to receive a cumulative record per document that is close to the full metadata record.

7.2.2 Data Integration

Having a diverse range of content providers to HDML, which additionally utilizes different data schemas, the data aggregated require integration in order to appear unified and accessible through search functions. Accordingly, HDML efforts target the creation of an HDML data schema that will be able to accommodate all such information in the best possible manner.

7.2.3 Data Provision to EuDML

HDML, in order to provide its full potential of outreach, needs to ensure seamless and automated data dissemination to EuDML for the current content as well as for any future content submitted to its repositories.

For this to be possible, methods are being implemented in order to ensure prototyped methodology of new data insertion to HDML as well as automated interconnectivity to EuDML, especially for potential data updates.

7.3 IMI-BAS's Cyrillic OCR

IMI-BAS has trained Tesseract to recognize the Cyrillic alphabet (Bulgarian). The Bulgarian trained data file can be downloaded from http://code.google.com/p/tesseract-ocr/downloads/detail?name=bul.traineddata.gz&can=2&q=

7.4 Commercial Services and Tools

7.4.1 Portico

Portico offers publishers and libraries a wide range of services, including:

- To analyze the formats and packaging of sample or supplied content.
- Guided by Portico's established preservation policies, the develop a preservation plan appropriate to the supplied content and the needs of the content provider and library participants in the service. The preservation plan may include an initial migration of the packaging or files in specific formats (for example, Portico migrates publisher specific e-journal article XML to the NLM archival standard).
- To develop any new tools necessary to implement the plan. Possibilities include tools to:
  - retrieve the content;
  - load the content into the processing system (there is significant amount of *loaders* and applications set up to ingest and convert the data from publishers);
  - repackage the content;
  - migrate files in specific formats.

In this sense, most potential EuDML content providers are already paying for Portico's conversion and unification into NLM, as the main mathematics publishers[3] provide their born-digital data to Portico. It would significantly improve EuDML services and smooth out EuDML startup if math publishers with Portico would agree that EuDML could pick up the NLM unified articles for its service.

7.4.2 Datapage

Datapage has been providing the following XML related services:

- Generation of NLM DTD and submission of final valid and well-formed XML files to PubMed Central
- Generation and submission of XML metadata to DOAJ (Directory of Open Access Journals)
- Submission of DOIs and links to citations in CrossRef
- Full text XML submissions to archive systems such as Portico

Athena [20] is an XML authoring tool developed in-house at Datatype, is a core component of all data related activity within the company. This is used by the teams for styling, structuring and conversion to XML conforming to specific DTDs. A basic XML file is created at a very early stage in the workflow and the automated conversion of basic XML to XML DTD required is then performed by the tool. During the final conversion stage, metadata is extracted from any given full text XML file and validated

3. that publish more than half of mathematical content

against the specific DTD requirements. The tool can also be customized to changing DTD requirements such as inclusion of new elements, attributes, entities, etc.

8 Summary, Conclusions

We have presented a wide range of techniques, tools and experience related to the task of metadata enhancements. We have concentrated on available tools that may be used for EuDML toolsets [58, 39]. We have identified expertize among project partners and outlined possible workflows to be used for EuDML enhancers.

References

[1] Adobe Systems Incorporated. *Adobe Systems Incorporated: PDF Reference*, pages 90–100. Adobe Systems Incorporated, sixth edition, 2006. http://www.adobe.com/devnet/acrobat/pdfs/pdf_reference_1-7.pdf.

[2] R. H. Anderson. *Syntax-Directed Recognition of Hand-Printed Two-Dimensional Mathematics*. PhD thesis, Harvard University, January 1968.

[3] Kristin Antelman. Do Open-Access Articles Have a Greater Research Impact? *College & Research Libraries*, 65(5):372–382, September 2004. http://www.ala.org/ala/mgrps/divs/acrl/publications/crljournal/2004/sep/antelman.pdf.

[4] Josef B. Baker, Alan P. Sexton, and Volker Sorge. Extracting Precise Data from PDF Documents for Mathematical Formula Recognition. In *DAS 2008: Proceedings of The Eighth IAPR Workshop on Document Analysis Systems*, 2008.

[5] Josef B. Baker, Alan P. Sexton, and Volker Sorge. A Linear Grammar Approach to Mathematical Formula Recognition from PDF. In *Proceedings of the 8th International Conference on Mathematical Knowledge Management in the Proceedings of the Conference in Intelligent Computer Mathematics*, volume 5625 of *LNAI*, pages 201–216, Grand Bend, Canada, July 10–12 2009. Springer.

[6] Josef B. Baker, Alan P. Sexton, and Volker Sorge. Faithful mathematical formula recognition from PDF documents. In *DAS '10: Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, pages 485–492, New York, NY, USA, 2010. ACM.

[7] Miroslav Bartošek, Petr Kovář, and Martin Šárfy. DML-CZ Metadata Editor: Content Creation System for Digital Libraries. In Sojka [56], pages 139–151. http://www.fi.muni.cz/~sojka/dml-2008-program.xhtml.

[8] Miroslav Bartošek, Martin Lhoták, Jiří Rákosník, Petr Sojka, and Martin Šárfy. DML-CZ: The Objectives and the First Steps. In Jonathan Borwein, Eugénio M. Rocha, and José Francisco Rodrigues, editors, *CMDE 2006: Communicating Mathematics in the Digital Era*, pages 69–79. A.K. Peters, MA, USA, 2008.

[9] Dan Bloomberg. Leptonica. [online], 2010. http://www.leptonica.com/.

[10] José Borbinha, Wojtek Sylwestrzak, Gilberto Pedrosa, and Aleksander Nowinski. EuDML Global System Functional Specification, November 2010. Deliverable D4.1 of EU CIP-ICT-PSP project 250503 EuDML: The European Digital Mathematics Library, http://eudml.eu/.

[11] Thierry Bouche. A pdfLaTeX-based automated journal production system. *TUGboat*, 27(1):45–50, 2006.

[12] Thierry Bouche. CEDRICS: When CEDRAM Meets Tralics. In Sojka [56], pages 153–165. http://dml.cz/dmlcz/702544.

[13] Thierry Bouche. Towards a Digital Mathematics Library? In Jonathan Borwein, Eugénio M. Rocha, and José Francisco Rodrigues, editors, *CMDE 2006: Communicating Mathematics in the Digital Era*, pages 43–68. A.K. Peters, MA, USA, 2008.

[14] Thierry Bouche and Hugo Manguinhas. Report on available collections and metadata, November 2010. Deliverable D3.1 of EU CIP-ICT-PSP project 250503 EuDML: The European Digital Mathematics Library, http://eudml.eu/.

[15] Pascal Chevalier. i2S DigiBook Mag, issue no. 2, July 2002. http://ww.i2s-bookscanner.com/pdf/digibook_mag_no2.pdf.

[16] P. A. Chou. Recognition of equations using a two-dimensional stochastic context-free grammar. In *SPIE Conference on Visual Communications and Image Processing*, pages 852–865, November 1989.

[17] JBIG Committee. *14492 FCD*. ISO/IEC JTC 1/SC 29/WG 1, 1999. http://www.jpeg.org/public/fcd14492.pdf.

[18] Silviu Cucerzan and David Yarowsky. Language independent, minimally supervised induction of lexical probabilities. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 270–277, Morristown, NJ, USA, 2000. Association for Computational Linguistics.

[19] CVISION Technologies, Inc. Impact: Improving access to text, 2010. http://www.impact-project.eu/taa/tech/?type=98.

[20] Datatype, Inc. Athena, 2010. http://www.datapage.ie/athena.html.

[21] DML-CZ. Digitization metadata editor. http://sourceforge.net/projects/dme/, 2009.

[22] DML-CZ: The Czech Digital Mathematics Library. http://dml.cz/, http://project.dml.cz/.

[23] Umberto Eco. Vegetal and mineral memory: The future of books. Al-Ahram, Issue No. 665, November 2003. http://weekly.ahram.org.eg/2003/665/bo3.htm.

[24] Ray Smith et al. tesseract-ocr. http://code.google.com/p/tesseract-ocr/.

[25] Yuko Eto and Masakazu Suzuki. Mathematical formula recognition using virtual link network. In *ICDAR '01: Proceedings of the Sixth International Conference on Document Analysis and Recognition*, page 762, Washington, DC, USA, 2001. IEEE Computer Society.

[26] Richard Fateman. How to Find Mathematics on a Scanned Page. In Daniel P. Lopresti and Jiangying Zhou, editors, *Documents of Recognition and Retrieval VII*, volume 3967, pages 98–109, December 1999.

[27] Richard J. Fateman and Taku Tokuyasu. Progress in Recognizing Typeset Mathematics. *Proceedings of SPIE — The International Society for Optical Engineering*, 2660:37–50, 1996. http://citeseer.ist.psu.edu/article/fateman96progress.html.

[28] Richard J. Fateman, Taku Tokuyasu, Benjamin P. Berman, and Nicholas Mitchell. Optical Character Recognition and Parsing of Typeset Mathematics. *J. Visual Communication and Image Representation*, 7(1):2–15, 1996. http://citeseer.ist.psu.edu/fateman96optical.html.

[29] A. Grbavec and D. Blostein. Mathematics recognition using graph rewriting. In *ICDAR '95: Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1)*, page 417, Washington, DC, USA, 1995. IEEE Computer Society.

[30] Eitan Gurari. TEX4ht. http://www.tug.org/applications/tex4ht/mn.html.

[31] Radim Hatlapatka and Petr Sojka. PDF Enhancements Tools for a Digital Library: pdfJbIm and pdfsign. In Sojka [57], pages 45–55. http://is.muni.cz/publication/891674/.

[32] John D. Hobby. Matching document images with ground truth. *International Journal on Document Analysis and Recognition*, 1:52–61, 1998.

[33] John D. Hobby and Tin Kam Ho. Enhancing degraded document images via bitmap clustering and averaging. In *ICDAR '97: Proceedings of the 4th International Conference on Document Analysis and Recognition*, pages 394–400, Washington, DC, USA, 1997. IEEE Computer Society.

[34] Michael Jost, Thierry Bouche, Claude Goutorbe, and Jean-Paul Jorda. The EuDML metadata schema, November 2010. Deliverable D3.2 of EU CIP-ICT-PSP project 250503 EuDML: The European Digital Mathematics Library, `http://eudml.eu/`.

[35] Adam Langley. Homepage of jbig2enc encoder. [online]. `http://github.com/agl/jbig2enc`.

[36] S. Lavirotte and L. Pottier. Mathematical Formula Recognition Using Graph Grammar. In *Proceedings of the SPIE, Document Recognition V*, volume 3305, pages 44–52, San Jose, CA, USA, 1998. `http://citeseer.ist.psu.edu/lavirotte98mathematical.html`.

[37] Steve Lawrence. Online or invisible? *Nature*, 411(6837):521, 2001.

[38] Mark Lee, Petr Sojka, Volker Sorge, Josef Baker, Wojtek Hury, and Lukasz Bolikowski. Association Analyzer Implementation: State of the Art, November 2010. Deliverable D8.1 of EU CIP-ICT-PSP project 250503 EuDML: The European Digital Mathematics Library, `http://eudml.eu/`.

[39] Mark Lee, Petr Sojka, Volker Sorge, Josef Baker, Wojtek Hury, Lukasz Bolikowski, and Radim Řehůřek. Toolset for Entity and Semantic Associations – Initial Release, January 2011. Deliverable D8.2 of EU CIP-ICT-PSP project 250503 EuDML: The European Digital Mathematics Library, `http://eudml.eu/`.

[40] Filej Miha, Michal Růžička, Martin Šárfy, and Petr Sojka. Metadata Editing and Validation for a Digital Mathematics Library. In Sojka [57], pages 57–62. `http://is.muni.cz/publication/891710/`.

[41] Digital Archive of Journal Articles National Center for Biotechnology Information (NCBI) and National Library of Medicine (NLM). Journal archiving and interchange tag set tag library version 3.0, November 2008. Developed by Mulberry Technologies, Inc., available at `http://dtd.nlm.nih.gov/archiving/tag-library/`.

[42] Nasayuki Okamoto and Bin Miao. Recognition of Mathematical Expressions by Using the Layout Structures of Symbols. In *Proc. of ICDAR '91*, pages 242–250, 1991.

[43] Martin Proulx. A solution to mathematics parsing. `http://www.cs.berkeley.edu/~fateman/papers/pres.pdf`, April 1996.

[44] Tomáš Pulkrábek. Obrazové transformace při digitalizaci textů (in Czech, Image Transformation during Digitisation). Master's thesis, Faculty of Informatics, 2008. Bachelor Thesis Masaryk University, Brno, Faculty of Informatics, `https://is.muni.cz/th/139908/fi_b/?lang=en`.

[45] Amar Raja, Matthew Rayner, Alan Sexton, and Volker Sorge. Towars a Parser for Mathematical Formula Recognition. In Jonathon M. Borwein and William M. Farmer, editors, *Proceedings of 5th International Conference, MKM 2006*, volume 4108 of *Lecture Notes in Computer Science*, pages 139–151. Springer, August 2006.

[46] Yves Rangoni, Faisal Shafait, and Thomas M. Breuel. OCR Based Thresholding. In *Proceedings of MVA 2009 IAPR Conference on Machine Vision Applications*, pages 3–18, May 2009.

[47] Radim Řehůřek and Petr Sojka. Automated Classification and Categorization of Mathematical Knowledge. In Serge Autexier, John Campbell, Julio Rubio, Volker Sorge, Masakazu

Suzuki, and Freek Wiedijk, editors, *Intelligent Computer Mathematics—Proceedings of 7th International Conference on Mathematical Knowledge Management MKM 2008*, volume 5144 of *Lecture Notes in Computer Science LNCS/LNAI*, pages 543–557, Berlin, Heidelberg, July 2008. Springer-Verlag.

[48] Michael Riley. OpenFst Library. http://www.openfst.org/.

[49] Michal Růžička. Automated Processing of TEX-typeset Articles for a Digital Library. In Sojka [56], pages 167–176. http://www.fi.muni.cz/~sojka/dml-2008-program.xhtml.

[50] Michal Růžička and Petr Sojka. Data Enhancements in a Digital Mathematics Library. In Sojka [57], pages 69–76. http://dml.cz/dmlcz/702575.

[51] Sunita Sarawagi. Information extraction. *Found. Trends databases*, 1:261–377, March 2008. http://portal.acm.org/citation.cfm?id=1498844.1498845.

[52] Alan P. Sexton and Volker Sorge. A database of glyphs for ocr of mathematical documents. In Michael Kohlhase, editor, *MKM*, volume 3863 of *Lecture Notes in Computer Science*, pages 203–216. Springer, 2005.

[53] Steven J. Simske and Xiaofan Lin. Creating Digital Libraries: Content Generation and Re-Mastering. In *Proceedings of First International Workshop on Document Image Analysis for Libraries (DIAL 2004)*, page 13, 2004. http://doi.ieeecomputersociety.org/10.1109/DIAL.2004.1263235.

[54] Ray Smith, Chris Newton, and Phil Cheatle. Adaptive Thresholding for OCR: A Significant Test. Technical Report HPL-1993-22, HP Laboratories Bristol, March 1993.

[55] Petr Sojka. From Scanned Image to Knowledge Sharing. In Klaus Tochtermann and Hermann Maurer, editors, *Proceedings of I-KNOW'05: Fifth International Conference on Knowledge Management*, pages 664–672, Graz, Austria, June 2005. Know-Center in coop. with Graz Uni, Joanneum Research and Springer Pub. Co.

[56] Petr Sojka, editor. *Towards a Digital Mathematics Library*, Birmingham, UK, July 2008. Masaryk University. http://www.fi.muni.cz/~sojka/dml-2008-program.xhtml.

[57] Petr Sojka, editor. *Towards a Digital Mathematics Library*, Paris, France, July 2010. Masaryk University. http://www.fi.muni.cz/~sojka/dml-2010-program.html.

[58] Petr Sojka, Josef Baker, Alan Sexton, Volker Sorge, and Radim Hatlapatka. Toolset for Image and Text Processing and Metadata Editing – Initial Release, January 2011. Deliverable D7.2 of EU CIP-ICT-PSP project 250503 EuDML: The European Digital Mathematics Library, http://eudml.eu/.

[59] Petr Sojka and Radim Hatlapatka. Document Engineering for a Digital Library: PDF recompression using JBIG2 and other optimization of PDF documents. In *Proceedings of the ACM Conference on Document Engineering, DocEng 2010*, pages 3–12, Manchester, September 2010. Association of Computing Machinery. http://portal.acm.org/citation.cfm?id=1860563.

[60] Volker Sorge, Alan Sexton, Josef Baker, and Petr Sojka. State of the Art of Accessibility Tools, January 2011. Deliverable D10.1 of EU CIP-ICT-PSP project 250503 EuDML: The European Digital Mathematics Library, http://eudml.eu/.

[61] Masakazu Suzuki. Infty Project. http://www.inftyproject.org/.

[62] Masakazu Suzuki, Fumikazu Tamari, Ryoji Fukuda, Seiichi Uchida, and Toshihiro Kanahori. INFTY: an integrated OCR system for mathematical documents. In *DocEng '03: Proceedings of the 2003 ACM symposium on Document Engineering*, pages 95–104, New York, NY, USA, 2003. ACM Press. http://doi.acm.org/10.1145/958220.958239.

[63] The Unicode Consortium. *The Unicode Standard, Version 5.0.* Addison-Wesley, Reading, MA, USA, 2006. www.unicode.org/unicode/standard/standard.html.

[64] Xuedong Tian and Haoxin Fan. Structural analysis based on baseline in printed mathematical expressions. In *PDCAT '05: Proceedings of the Sixth International Conference on Parallel and Distributed Computing Applications and Technologies*, pages 787–790, Washington, DC, USA, 2005. IEEE Computer Society. http://dx.doi.org/10.1109/PDCAT.2005.228.

[65] Seiichi Toyota, Seiichi Uchida, and Masakazu Suzuki. Structural analysis of mathematical formulae with verification based on formula description grammar. In *DAS 2006: Proceedings of Document Analysis Systems VII, 7th International workshop*, volume 3872 of *Lecture Notes In Computer Science*, pages 153–163. Springer, 2006.

[66] International Telecomunication Union. *ITU-T Recommendation T.88*. ITU-T Recommendation T.88, 2000. http://www.itu.int/rec/T-REC-T.88-200002-I/en.

[67] Z. Wang and C. Faure. Structural analysis of handwritten mathematical expressions. In *Proc. of ICPR-9*, 1988.

[68] Michael Yang and Richard Fateman. Extracting mathematical expressions from postscript documents. In *ISSAC '04: Proceedings of the 2004 International Symposium on Symbolic and Algebraic Computation*, pages 305–311, New York, NY, USA, 2004. ACM Press. http://doi.acm.org/10.1145/1005285.1005329.

[69] Katarzyna Zamłyńska, Łukasz Bolikowski, and Tomasz Rosiek. Migration of the Mathematical Collection of Polish Virtual Library of Science to the YADDA platform. In Sojka [56], pages 127–130. http://dml.cz/dmlcz/702538.

[70] Richard Zanibbi. Recognition of mathematics notation via computer using baseline structure. Technical report, Department of Computing and Information Science Queen's University, Kingston, Ontario, Canada, August 2000.

[71] Richard Zanibbi, Dorothea Blostein, and James R. Cordy. Recognizing mathematical expressions using tree transformation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(11):1455–1467, 2002. http://dx.doi.org/10.1109/TPAMI.2002.1046157.

Index

ABBYY, 11
analysis, 13
any data related, 4
AnyDoc Software, 9
automatic, 3

below-image, 8
bibTeX, 26
binarisation, 7
blur filter, 7
Book Restorer, 8
Brainware, 9
BWMeta format, 27

CEDRAM, 26
CEDRICS, 26
clues, 14
coreference resolution, 14
cropping, 6
CuneiForm, 9

DeskLight, 27
despeckle filter, 7
disambiguation, 14
DML-CZ, 8, 28, 29
document, 4

entity recognition, 13, 14
ePub, 23
EuDML core engine, 3
EuMDR, 24
eutools, 5
evidences, 14
ExperVision, 9

features, 14
filter
    blur, 7
    despeckle, 7
    Sigma, 7
    Wiener, 7
FineReader, 9

Gallica-MATH, 26
Gellmu, 22
geometrical correction, 6
GOCR, 9

Hevea, 22
HTeX, 22
HTMX, 22
HyperLaTeX, 22
HyperTeX, 22

image capture, 6
improved jbig2enc, 28
information models, 24
Infty, 9
is worth optimizing, 8

JBIG2, 8
Jbig2enc, 28

LaTeX2HTML, 22
LaTeX2hyp, 22
LaTeX2SGML, 22
leptonica, 28
library
    DML-CZ, 8, 28, 29
    improved jbig2enc, 28
    leptonica, 28
loaders, 31
Ltoh, 22
Ltx2x, 22

mappings, 24
math indexing and search, 3
Math2HTML, 22
MathML, 15
MDR, 24
metadata enhancements, 4
Metadata Registry, 24
mini-DML, 26

NLM, 23
NLM Archiving DTD Suite, 3
Noscitur a sociis, 14
NUMDAM, 26

Ocaml, 23
OCR of mathematics, 3
Ocrad, 9
OCRopus, 9
OMDoc, 15
Omega, 22
Omnipage, 9
OpenFST, 10
OpenMath, 15
Optical Character Recognition, 8
optical character recognition (OCR), 6

pdfJbim, 28

pdfsign, 28
pdfsizeopt.py, 28
perceptually lossless, 8
project
  CEDRAM, 26
  Gallica-MATH, 26
  mini-DML, 26
  NUMDAM, 26
publish/export, 7

Readiris, 9
ReadSoft, 9
relationship extraction, 14

services, 5
sets, 24
Sigma, 7
Simple OCR, 9
SmartScore, 9

TakOCR, 9
TechExplorer, 22
terminology extraction, 14
Tesseract, 9
TeX2HTML, 22
TeX2RTF, 22
TeX4ht, 22
Texi2HTML, 22
TeXpider, 22
the, 4
tool
  bibTeX, 26
  Book Restorer, 8
  CEDRICS, 26
  DeskLight, 27
  EuDML core engine, 3
  Gellmu, 22
  Hevea, 22
  HTeX, 22
  HTMX, 22
  HyperLaTeX, 22
  HyperTeX, 22
  Jbig2enc, 28
  LaTeX2HTML, 22
  LaTeX2hyp, 22
  LaTeX2SGML, 22
  Ltoh, 22
  Ltx2x, 22
  Math2HTML, 22
  OCR
    AnyDoc Software, 9
    Brainware, 9
    CuneiForm, 9
    ExperVision, 9
    FineReader, 9
    GOCR, 9
    Infty, 9
    Ocrad, 9
    OCRopus, 9
    Omnipage, 9
    Readiris, 9
    ReadSoft, 9
    Simple OCR, 9
    SmartScore, 9
    Tesseract, 9
  Omega, 22
  OpenFST, 10
  pdfJbim, 28
  pdfsign, 28
  pdfsizeopt.py, 28
  TakOCR, 9
  TechExplorer, 22
  TeX2HTML, 22
  TeX2RTF, 22
  TeX4ht, 22
  Texi2HTML, 22
  TeXpider, 22
  Tralics, 22, 26
  Tth, 22
  Vulcanize, 22
  WebEq, 22
toolsets, 5
Tralics, 22, 26
Tth, 22

virtual, 3
Vulcanize, 22

WebEq, 22
Wiener, 7
word sense disambiguation, 14
WSD, 14