Document Engineering for a Digital Library

PDF recompression using JBIG2 and other optimization of PDF documents

Petr Sojka
Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
sojka@fi.muni.cz

Radim Hatlapatka
Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
208155@mail.muni.cz

ABSTRACT

This paper describes several innovative document transformations and tools that have been developed in the process of building the Digital Mathematical Library DML-CZ `http://dml.cz`. The main result presented in this paper is our PDF re-compression tool, developed using a jbig2enc library. Together with other programs, especially pdfsizeopt.py by Péter Szabó, we have managed to decrease PDF storage size and transmission needs by 62%: using both programs we reduced the size of the original PDFs to 38%.

This paper briefly describes other approaches and tools developed while creating the digital library. The batch digital signature stamper, the document similarity metrics which uses four different methods, a [meta]data validation process and some math OCR tools represent some of the main byproducts of this project. These ways of document engineering, together with Google Scholar indexing optimization, have led to the success of serving digitized and born-digital scientific math documents to the public in DML-CZ, and will be employed also in the project of The European Digital Mathematics Library, EuDML.

Categories and Subject Descriptors

H.3.7 [**Information Systems**]: Information storage and Retrieval—*Digital Libraries*; I.7 [**Computing Methodologies**]: Document and text Processing; H.3.2 [**Information Systems**]: Information storage and Retrieval—*Information storage*; H.3.6 [**Information Systems**]: Information storage and Retrieval—*Library Automation*; J.7 [**Computer Applications**]: Computers in other systems—*Publishing*

General Terms

Algorithms, Design, Experimentation, Performance, Security

Keywords

Authoring tools and systems; Categorization; Classification; Document presentation (typography, formatting, layout); Representations/Standards; Structure, layout and content analysis; Character recognition; Digital mathematical library, Digitisation workflow

DocEng2010, September 21–24, 2010, Manchester, United Kingdom.

1. MOTIVATION AND DESIGN

> Pyramids, cathedrals, and rockets exist not because of geometry, theory of structures, or thermodynamics, but because they were first pictures—literally visions—in the minds of those who conceived them.
> (Eugene Ferguson [12])

A digital library (DL) business needs the art and craft of document engineering in many ways. A well designed DL will necessarily tackle issues of scalability, persistence, transfer size and speed, linking related data, format migration, data curatorship, visibility, etc. Process-oriented *workflows* enact the machinery of building and running the digital library and its *services*.

People dream of systems that will help to collect, store and serve verified scientific knowledge in relevant thematic areas: PubMed Central (PMC) is one such succesful system in the medical domain. A realization of the dream of a World Digital Mathematics Library [15] is yet to come.

We report on the experiences gained, lessons learned and the tools prepared for document engineering for The Czech Digital Mathematics Library DML-CZ project and which are now being adapted for The European Digital Mathematics Library, EuDML [41]. The aim of the DML-CZ project funded from 2005 to 2009, was to digitize the relevant mathematical literature published in the Czech lands. It comprises periodicals, selected monographs and conference proceedings from the nineteenth century up to and including the most recently produced mathematical publications [34, 36, 37]. It has been launched and is readily available at dml.cz, serving almost 30,000 articles on 300,000 pages to the public.

The general workflow of the project has been designed [31] in close collaboration with mathematicians and librarians, as modular and extensible. It was iteratively developed and about dozen of new tools was designed, implemented and deployed, as needed to realize functional specification. Rather than monolithic uniform system, DML-CZ evolved as loosely coupled system of tools and interfaces. As shown in Figure 1, workflow reflects the different types of acquired heterogeneous 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 an automated export of data for the digital library.

To achieve demanded functionality, metadata has to be made uniform via complex workflow.

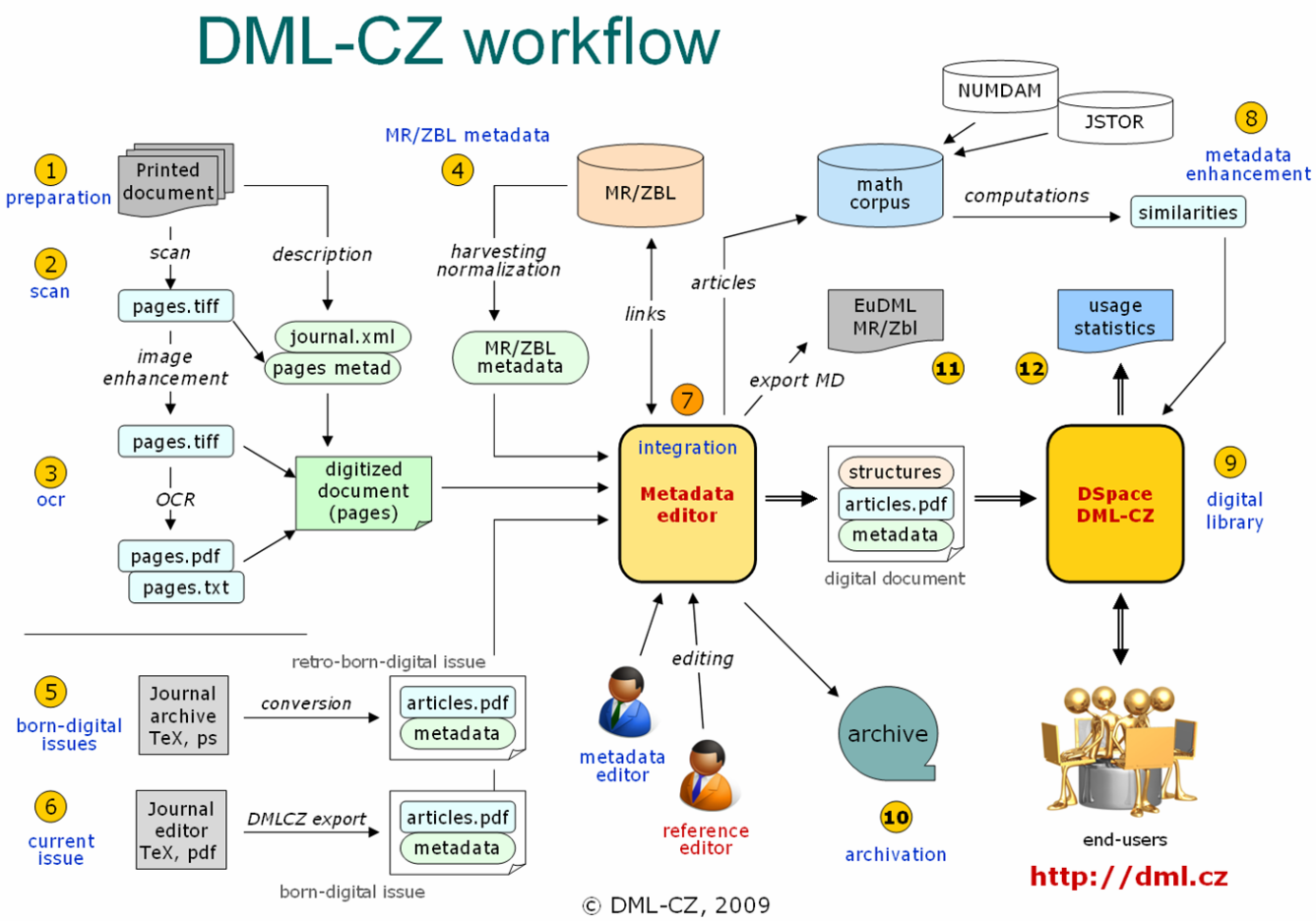

Figure 1: DML-CZ top-level workflow scheme

Within the project, several general purpose tools have been developed:

1. scripting of transformation pipes of scanned images [24],
2. DML-CZ OCR workflow allowing recognition of scanned mathematical documents [35],
3. web-based Metadata Editor [2],
4. tools for classifying mathematical documents and measuring their similarity [25];
5. workflow for born-digital publication production with direct export of metadata for DML [27];
6. extension to LUCENE engine allowing the indexing of mathematics;
7. batch PDF stamper for digital signing of PDFs produced;
8. PDF optimizer re-compressing image objects in PDF with the new JBIG2 compression filter supported by Adobe since PDF specification version 1.4 (Adobe Reader 5); and
9. batch article PDF generation with titlepage by XeLaTeX [28].

In the following sections we describe a part (steps 2, 3, 7 and 8 in Figure 1) of our digitisation workflow in the hope that they can be used by similar projects or even in other domains. Some tools, such as the PDF re-compressor, have a more general application with a possible wider impact beyond our digital libraries.

2. OPTICAL CHARACTER RECOGNITION – DML-CZ OCR

> The road to wisdom? Well, it's plain and simple to express:
> Err and err and err again, but less and less and less. (Piet Hein)

In order to index papers, it is necessary to obtain a full text from the page bitmaps through optical character recognition. Also, we need to recognize page numbers located in every TIFF to link the page images to the article metadata.

Tests with various OCR programmes showed that not one of them gives acceptable results for mathematical content, with character error rates often exceeding 10% (counting wrong character positions and font types as errors). For text recognition, FINEREADER by ABBYY® gives the best results, whereas for the structural recognition of mathematics, InftyReader [40] gives impressive results.

The FINEREADER software development kit (SDK for Windows version 8.1) was used to develop a part of the system for the location and recognition of page numbers. The batch system, DML-CZ OCR [32, 38], takes sequences of TIFF images and produces two-layer single page PDFs with invisible full-texts behind the images. The processing starts with the recognition of languages used in each paragraph, and then blocks are recognized again with a

special setting (using language dictionaries) for every given block of the text. With such a fine-tuning of parameters, we are able to achieve a character error rate of a mere one percent [38].

Of the solutions and software evaluated on plain texts, FINEREADER gave the best results, but it has no support for the recognition of mathematical expressions. Texts without recognized maths may be sufficient for basic indexing and searching. However, it is not surprising that omitting maths is of considerable significance when the full texts are used for such tasks as automated text classification and categorization or for computing paper similarity [39]. Therefore we are striving to enhance the state-of-the-art possibilities for mathematical OCR.

Neither ABBYY® nor Google® responded positively to the near future of the math OCR development plans—mathematics is only a small niche market for them. On the other hand, developers of the INFTYREADER system [40] were willing to gradually improve their support for European languages, MATHML and LaTeX export filters and to enrich their database of mathematical symbols.

We have found that setting the parameters of the OCR engine (language, word-list consultation) influences the precision significantly. We trained FINEREADER on the type cases used at the printer's where the journals were typeset.

At the end of these extensive experiments, we developed a method of OCR processing consisting of several phases, both in FINEREADER and in Infty. Processing using FINEREADER involves the following steps:

1. A page or block of text is recognized for the first time using a universal setup (non-language specific). A histogram of character bigrams and trigrams from words with lengths greater than three is created.

2. The computed histogram of the text block is compared [11] to the histograms created from the journal data during the training phase for all languages used: English, French, Russian, German and Czech. The perl module `Lingua::Ident` is used. The block containing the bibliography is detected by different algorithms and is treated differently.

3. A page or block of text is processed for the second time with parameters optimized for the 'language' recognized in the previous step and saved as a two-layer PDF (with a text layer used for searching, indexing and similarity computation).

The recognition of mathematical formulae in FINEREADER is not satisfactory, however. The only suitable tool for this domain that we have encountered and experimented with is INFTY. INFTY's new PDF import capability has manifold ramifications: it will allow the import of our current FINEREADER's two-layer PDFs, the use of the text part only, the rejection of poorly recognized maths and both the detection and recognition of maths expressions. A new INFTY version that combines FINEREADER's technology (OCR voting [21]) is in preparation. In the meantime,

1. PDF is passed to INFTYREADER and the results are stored in the INFTY Markup Language (IML) and in LaTeX (Human readable LaTeX).

2. IML is postprocessed by a home-grown programme in JAVA to fix the recognition errors of some of the accented characters that INFTY does not yet have in its glyph database.

Using the process outlined above, we have managed to decrease the character error rate from an initial 11.35% (a universal language setup of FineReader) to an average 0.98% character error rate. [23, 22, 33] The whole processing is fully automated after the initial font recognition and language detection training. The error rate may be further decreased when INFTY's character database is semi-automatically enriched while processing a new journal.

3. HANDLING METADATA

> Only if every user has a common and exact understanding of the data can it be exchanged trouble-free.
> (ISO/IEC 11179 Metadata Registry Specification)

It is well known that providing complete, correct and reliable metadata is very time-consuming. The information-rich metadata needed for a full-featured digital mathematics library consists of more than the standard sets of metadata corresponding to the Dublin Core Metadata Element Set; the bibliographical references are of importance as are language alternations of titles, different spellings of names and full-texts obtained and indexed by OCR.

It is essentially impossible to anticipate all questions and problems that may appear during the digitisation of mathematical literature, especially older texts. The multilingual content of the DML-CZ only exacerbates the problem. Every paper is provided with the original title (except for Russian ones) and with its English translation. We add additional language versions of the title whenever they are available. This happens, for example, when the original paper is in Czech or Russian and there is a corresponding German or French entry in Zentralblatt MATH. We keep all these versions in the metadata.

Most mathematical journals require authors to classify their papers by one primary and possibly several secondary codes of the Mathematics Subject Classification (MSC) today. Providing missing MSC codes for old papers, prior to the adoption of this system, poses a particular problem. The task of assigning MSC codes for retro-digitized articles requires qualified mathematicians. Terminology evolved and understanding certain expressions requires reading and understanding the whole paper in the context of its time. Help is at hand in the guise of MSC codes suggested by an automated classifier trained via machine learning techniques from a database of articles already classified [39, 25].

The references are presented in the original languages which in fact means that the list of references for a single paper may include any combination of Czech, Slovak, English, French, Russian, German, Italian. Even though we use our OCR techniques with their automated identifications of the block of references and relatively reliable language detection, manual control and correction remains necessary.

Harvesting the metadata from Zentralblatt MATH and Mathematical Reviews is of considerable value, although several types of problems still appear: when the paper is not written in English and the English translations of titles in the two databases differ, the question of which one should be accepted arises. Furthermore, correcting an unsatisfactorily translated title is contentious. And a similar problem concerns authors' names which arises through different transcription preferences as seen frequently in Russian, Chinese and Vietnamese names. Lastly, there is the matter of MSC codes.

Starting with the *Czechoslovak Mathematical Journal* as the pilot project, we have designed a workflow and developed several tools to handle the metadata at minimum cost. However, as we proceed, especially to the older journals, it was clear that the tools needed to be developed still further.

The most important tool we use in this respect is the *Metadata Editor* [2, 9] which has gradually developed into an efficient web application that allows simultaneous distant editing according to assigned structured access rights. It supports two levels of actions. On the first, the operators editing the data are provided with page

thumbnails so that they can check the completeness, scan the quality and configuration of the articles, shuffle the pages easily and cut or merge articles if necessary. On the other level, the operator can check the automatically imported metadata, edit and complete them. An important integral part of the Metadata Editor is the module that administers the 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. This represents a wider task which should be tackled with a concerted international effort. To assign various spellings of names to the proper person often requires an awareness of the author and of his or her work. The older the entry, the more difficult the task might be.

These functionalities in combination with remote access enable the distribution of the work among several people at different levels of expertise. Students of mathematics are usually employed to work at the first 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, and provide notes signalizing possible problems. Experienced mathematicians then add the necessary translations, complete the missing MSC codes, and provide links between related papers. They also arrive at the final revision with a validation of the metadata.

We consider bibliographical references important metadata of every paper. Their availability makes it possible to use professional systems like CrossRef for cross-publisher citation linking. The work starts with an 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. They are enhanced via computed links to Zentralblatt MATH, Mathematical Reviews and other sources [18]. The operator then checks, edits and approves the list of paper references.

For fixing errors that can be safely detected (as MSC code string invalid in MSC 2000) procedures are formulated and coded. They are automatically run as overnight jobs together with updates of the database and metadata statistics.

Finally, various detection procedures of possible errors have been suggested, evaluated and implemented for finding anomalous and suspicious content of metadata fields, with lists of generated warnings including hyperlinks for easy checking by an operator. An important control concerns the integrity of TEX sequences in metadata to assure a seamless typesetting of article cover pages in the later stage: all metadata to be typeset are exported in a single large file with a unique reference to the article, and typeset by XeLATEX to check the TEX control sequences used in the metadata fields.

Similar procedures allow for an efficient and economical increase of metadata completeness and quality.

4. METADATA FROM THE RETRO-BORN-DIGITAL PERIOD

Some of the journals we process come from earlier times when some kind of an electronic form was already available. For this reason, it is not necessary to scan, but on the other hand, a wide variety of formats and encodings creates its own challenges.

From the publisher of *Archivum Mathematicum* we obtained TEX sources. The publisher of *Czechoslovak Mathematical Journal*, *Applications of Mathematics*, and *Mathematica Bohemica*) provided us with a mixture of POSTSCRIPT, PDF and TEX files. Even the files for a single volume of a journal might not be homogeneous: TEX formats, macros and bibliography citations typesetting differ, so that developing a conversion filter for every file format and every markup is not an efficient solution.

We have developed several strategies for extracting a reliable citation list for every paper:

- generating from BIBTEX file;
- starting from LATEX's *thebibliography* environment grabbed from TEX file massaged by PERL script;
- starting from the plain text extracted from a POSTSCRIPT or a PDF file;
- rerunning the $\mathcal{AMS}$-TeX file with modified macros that write out tagged citations externally (used for *Commentat. Math. Univ. Carolinae*).

In the case when none of the above strategies is easily applicable, we resort to the standard OCR workflow starting from an electronic version of a page.

5. DATA FROM THE BORN-DIGITAL PERIOD

Following the idea of the CEDRAM project, we wanted to automate the import of newly published papers as much as possible. We cooperated with the editors of *Archivum Mathematicum* in developing the journal production workflow in the way that the data and metadata being imported into the DML-CZ are created as an automatic byproduct of the preparation process of the printed issue.

The new LATEX and BIBTEX style files that we have developed and all the citations for every paper are now stored in our new BIBTEX format. A set of conversion utilities and a workflow that extensively uses the programmes `make` (a UNIX tool which automates the generation and handling of dependencies), TRALICS (an XML converter), and JABREF (for BIBTEX citation management) have been developed and the the volumes of the journal since 2009 are the tangible products of this pilot project. The process is described in detail in [27].

6. DIGITAL PAPERS DELIVERY

In DML-CZ, we decided to support an article (book chapter) oriented delivery which is usually supported in scientific digital libraries such as Springer Link, as opposed to the page-oriented systems used in Göttingen Digitization Center (GDZ).

The generation of a deliverable PDF, for every paper or book chapter consists of the following steps:

- check that all paper metadata and digital objects have been approved by the person operating the Metadata Editor web application;
- generate a LATEX source file with metadata for title page typesetting;
- generate title page PDFs with XeLATEX. These pages contain the full paper citation, a persistent URL and a copyright notice;
- merge the title page PDF and individual PDF pages of the article into a single PDF;
- set the PDF security options for viewing, printing, cutting and pasting, etc.;
- optimize (linearize) the PDF object with the program `pdfopt` from a GHOSTSCRIPT software suite;
- digitally sign the PDF using a DML-CZ certificate. This allows a recipient of the PDF to verify that it originated from the DML-CZ project. This is a standard Public Key Infrastructure (PKI) approach.
- import the PDF into the digital library, where a file hash is computed and the counting of downloads starts.

7. TEXT POSTPROCESSING AND METADATA ENHANCEMENTS

When in doubt, use brute force. (Ken Thompson)

The OCR step is followed by further text processing, and its results are used for editing metadata and references.

7.1 Metadata Editor

The Metadata Editor (ME) [2, 9] has gradually developed into a fully-fledged and efficient web application, that allows simultaneous remote editing according to assigned structured access rights: `https://editor.dml.cz`. It supports two levels of actions. On the first, the operators editing the data are provided with page thumbnails so that they 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 operators can check the automatically imported metadata, and 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.

We consider the bibliographical references as important metadata of every paper. Their availability makes it possible to use professional systems like CROSSREF® for a cross-publisher citation linking. The work starts from the 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 strings which are invalid in the MSC 2000 standard, procedures are formulated and coded in a 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 which are useful for the management of the Metadata Editor 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 large 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 the MathML format smoothly. Similar procedures enable an efficient and economical increase of metadata completeness and quality.

7.2 Mathematical Document Classification and Categorization

The article full texts have many applications, e.g. for document classification and categorization. Fine document classification enables document filtering to reach higher precision in information retrieval systems such as DML. The most commonly used classification system today is the Mathematics Subject Classification (MSC) scheme (www.ams.org/msc/). We have developed an MSC classifier (guessed MSC) that is able to assign the top-level MSC for the retro-digitized articles. Our results convincingly demonstrate the feasibility of a machine learning approach to the classification of mathematical papers [25].

Another round of experiments was done with mathematical document similarity computation. We have collected a corpus of full texts with more than 40,000 articles (from DML-CZ and NUMDAM) and we have computed paper similarities using *tfidf* [29] and Latent Semantic Analysis (LSA) [8] and Random Projection methods. Methods use a Vector Space Model, first converting articles to vectors and then using the cosine of the angle between the two document vectors to assess their content similarity [20]. The difference between the methods is that while *tfidf* works directly over tokens, LSA first extracts concepts, then projects the vectors into this conceptual space where it only computes the similarity.

We are now able to show links to the closest document lists in the DML-CZ article landing pages to get feedback from authors and readers so that we can evaluate the metrics computed in this experiment. Given that we will enrich our full text mathematical corpus significantly (with data from JSTOR, ARXIV and other sources as planned), we hope that it will help to tackle plagiarism, too.

7.3 Indexing and Search

The digital library system we have chosen for running DML-CZ is the open source system, DSPACE, developed at Boston MIT. DSPACE is now actively supported and maintained by the DSPACE Consortium—it has a customizable layout layer MANAKIN, embedded OAI-PMH (Open Archive Initiative Protocol for Metadata Harvesting) server and a scalable indexing library, a persistent URL support for every object in the library; its complexity and maintenance is still manageable.

DSPACE uses LUCENE—an open source library for information retrieval (indexing and search), that can be enriched by the special handling of mathematical formulae in the future.

A big question is how to represent the maths for indexing. Should it be MATHML, or should we create a specially crafted term algebra allowing the incorporation of structural similarity measures and term unification? The results of the project ARXMLIV `kwarc.info/projects/arXMLiv/` for translating of ARXIV `arxiv.org/` to XML+MATHML can serve as a testbed not only for the XML approach, but also for the rigorous comparison of several indexing schemes that are being tested. [10]

Discussions about a persistent URL for our PDFs led to the semantically rich URL of the pattern `dmlcz/`*handle#−authors−title*. The mandatory part of the URL ends with the handle number; other URLs with this prefix will be redirected to the full one. Indexing robots will thus use words from the main article's metadata (author, title) giving them a greater weight for rank computation—the kind of special optimizations for Google[bot].

As part of the DSPACE installation there is also an OAI-PMH server. We have set it up so that the metadata exported comply with the suggested recommendation [5]. This permits the smoothest possible integration of metadata into larger virtual libraries of mathematics and even into WDML.

Further details of DSPACE customization for the DML-CZ web presentation `dml.cz` are described in [3] and [17].

8. INTRODUCTION TO JBIG2

JBIG2 is a standard for compressing bi-tonal images developed by Joint Bi-level Image Experts Group. These are images that consist of two colors only (usually black and white). The main area of such images is a scanned text. JBIG2 was published in 2000 as an international standard ITU T.88 [14] and one year later as ISO/IEC 14492 [16].

It typically generates files three to five times smaller than Fax Group 4 and two to four times smaller than JBIG1 (the previous standard released by Joint Bi-level Image Experts Group). JBIG2 supports lossy compression as well. It enables the compression ratio

to be increased several times without noticeable visual differences in comparison with lossless mode. Lossy compression without noticeable lossiness is called *perceptually lossless coding*. Scanned text often contains flyspecks (tiny pieces of dirt)—perceptually lossless coding can help get rid of the flyspecks and thus increase the quality of the output image.

8.1 Basic principles of JBIG2

The content of each page is segmented into several regions of which there are usually three—text region, halftone region and generic region. The text regions contain text, halftone regions contain halftone images[1] and generic regions contain the rest. In some situations, better results can be obtained by classifying text regions as generic ones and vice versa.

JBIG2 uses modified versions of adaptive Arithmetic and Huffman coding. Huffman coding is used mostly by faxes because of its lower computation demands. But Arithmetic coding gives slightly better results.

JBIG2 also supports multi-page compression that is used by symbol coding (coding of text regions). Any symbol that is frequently used on more than one page is stored in a global dictionary. Such symbols are thus stored just once and the space needed to store the documents is reduced.

8.2 JBIG2 in PDF

Since PDF version 1.4 (2001, Acrobat 5, see 3rd edition of the PDF Reference book) support for JBIG2Decode filter [1] has been embedded. This allows using compressed images according to standard JBIG2. This support has allowed the JBIG2 standard to spread far and wide without placing any burden on the end user.

PDF discards headers and some other data from JBIG2 images and puts this information into the PDF dictionary associated with the image object stream.

9. JBIG2ENC

Jbig2enc [19] is an open-source encoder developed with the support of Google by Adam Langley under an Apache License, Version 2.0. It uses Leptonica library [4] for image manipulation. Leptonica renders text, aligns symbol components and their comparison, among other things.

Jbig2enc supports only arithmetic coding and instead of halftone regions uses generic regions. It has embedded support for creating an output in a format suitable for PDF documents.

9.1 Modification of Jbig2enc

We are improving the perceptually lossless coding of the encoder, JBIG2ENC, which is used by a PDF re-compression tool (see Section 10).

At this point in its development we are trying to find accumulations of differences between pairs of symbols[2]. We look locally for accumulations in shapes of points or lines (vertical, diagonal or horizontal). When we find such an accumulation of differences, we consider these two symbols different. If we do not find any such accumulation of differences, we consider these symbols equivalent and unite them—all references to these templates are set to point to only one template and the second one is deleted.

To find such accumulations of differences we first apply the XOR operation to get an image bitmap containing differences only. Then we look at the subimages of this bitmap to see if it contains a significant enough accumulation of foreground pixels (different pixels of original images) to form a line or a point.

With this algorithm we are able to increase the compression ratio of the encoder JBIG2ENC by about eight percent, even for relatively low quality images.

We are now working on embedding OCR tools and techniques that will enhance the comparison process of two symbols. It should allow us to decrease the size of the output image to the size of the born-digital text.

As Figure 2 shows, the image before and after compression look the same at first sight despite the size of the compressed image being less than half of the original one. But they are not exactly the same. There are slight differences which are shown in the third image in Figure 2. Removing the slight differences between the same symbols would improve the quality of the output image.

Running JBIG2ENC removes some of the differences between the same symbols which can make the output quality appear either better or worse. It is crucial that the right representative that will stay in the text be chosen. To guarantee the improvement of quality, we always need to choose the best symbol from the equivalent ones. At this stage of development the first symbol is used as a representative symbol—it gives the same result as a random one, the quality remaining by and large the same.

10. PDF RE-COMPRESSOR

> It is this new world of complexity that is our peculiar domain.
> (Fred Brooks [6])

PDF re-compressor [13] is a tool written in Java which has been developed for DML-CZ. Its development is ongoing with the intention of it being used in EUDML. The main purpose is to decrease the size of PDF documents containing scanned text (mostly mathematical) and make it easier to transfer such documents via the Internet so that download time and costs can be reduced. PDF re-compressor even handles two-layer PDF documents.

The program replaces images with their re-compressed versions. It uses the JBIG2ENC encoder and two libraries written in Java: PDFBOX [43] and iText [7]. PDFBOX is used to extract raw image data and convert them to a suitable image format. IText is used for decrypting PDF documents if necessary and for replacing images with their re-compressed version. Information about the positions and dimensions of images is remembered during the process of extraction.

Jbig2enc allows the use of symbol coding that is suitable for scanned text. If symbol coding is employed, it segments images containing text to components and compares them. It is usually the case that one component contains exactly one symbol. All the same components are put in a dictionary after which only references to the dictionary are used.

A PDF re-compressor enables using a modified version of JBIG2ENC to achieve better results by enabling this option as an argument in a command line.

11. PDFSIZEOPT.PY

Pdfsizeopt.py [42] is a script written in Python under GNU General Public License. It combines different Unix tools and scripts to optimize the PDFs without damaging them. To optimize content streams, the recommended procedure is to first use one of the commercial optimizers, PDF Enhancer or Adobe Acrobat, and then run pdfsizeopt.py for optimizing mainly images and Type1 fonts. Pdfsizeopt.py also uses MULTIVALENT tool.pdf.Compress to do most

[1] You can find more about halftone at `http://en.wikipedia.org/wiki/Halftone`

[2] We work only with templates (representative symbols) returned by Leptonica

of the remaining work. If MULTIVALENT is installed, pdfsizeopt.py will run this automatically.

Pdfsizeopt.py also removes duplicate and unused data, serializes strings more effectively, compresses streams by high-effort ZIP, removes page thumbnails since they can be created whenever they are needed, etc. For these purposes it uses such tools as GHOSTSCRIPT, PDFTK, JBIG2ENC, PNGOUT, MULTIVALENT and PNG22PNM. GHOSTSCRIPT, for example, is used to convert fonts to CFF (Compact Font Format—Type 2, Type 1C). Then it unifies subsets of the same fonts.

For compression it uses high effort methods that are slower but more efficient (e.g. ZIP with PNGOUT). It runs several different compression methods suitable for each type of data and chooses the one with the best results. It supports JBIG2 compression by using the JBIG2ENC encoder (see Section 9) but it runs JBIG2ENC with generic coding, not symbol coding used for the text. By contrast, PDF re-compressor (see Section 10) does compress images using symbol coding and consequently achieves a better compression ratio for images containing text.

`Pdfsizeopt.py` uses some PDF specifics that have been used since version 1.5 which therefore requires Acrobat 6 or newer to view PDFs optimized by `pdfsizeopt.py`.

12. COMBINING PDFSIZEOPT.PY AND PDF RE-COMPRESSOR

If the computer scientist is a toolsmith, and if our delight is to fashion power tools and amplifiers for minds, we must partner with those who will use our tools, those whose intelligences we hope to amplify. (Fred Brooks [6])

To represent the results of each optimizer, we have applied them to data from the DML-CZ project. We used PDF documents from the journal, *Archivum Mathematicum*, which contains 6,641 pages in 665 papers from the years 1965 to 1991. These documents are two-layer documents, the OCR text being concealed below the scanned bitmap and used for indexing and searching.

In Tables 1 and 2 we present the results of running the optimizers. To retrieve statistical data we used `pdfsizeopt.py` with the option `--stats` before and after running the optimizer pipeline. The tables show how much is stored in each part of the document as well the size of the whole PDF document. All the values represented in the tables are counted as average values. Table 1 shows the results after running optimizers on PDF documents as single pages without any global dictionary being shared within pages. Table 2 shows the results for multi-page PDF documents comparing the effect of shared resources among pages in the PDF of a paper.

The most significant entries for making comparisons between `pdfsizeopt.py` and PDF re-compressor are *image objects* and *other objects*. This is because the PDF re-compressor does not optimize other parts of PDF documents. Images are stored in *image objects* but instead of being included in referenced objects, they are included in the section *other objects*. The global dictionary is a referenced object so it is counted as part of *other objects*.

Table 3 shows how much the size of PDF documents and the sizes of image data were reduced using the PDF re-compressor and pdfsizeopt.py in comparison with the original PDF. The results are represented as a percentage of the size of optimized PDFs in comparison with the original PDF file. It uses the same PDF corpus as in the previous table.

Image data in this case are counted together with the data of other objects because the global dictionary is stored in a separate object. The size of this object is counted in the section *other objects*. Unfortunately, the global dictionary is not the only thing stored as *other objects*. To at least partly distinguish which part of the data stored in other objects is a global dictionary, we use the size of *other objects* after running both optimizers for summarizing. For `pdfsizeopt.py` there is nothing more to optimize in the global dictionary data. By combining these two values, the results adequately indicate how effective each optimizer is for reducing the size of the image data.

As Table 3 shows the PDF re-compressor gives significantly better results than `pdfsizeopt.py` for images stored in multi-page PDF documents. We can see that ideal workflow is to run PDF re-compressor first and run `pdfsizeopt.py` on the result. It is mostly because of performance. In most cases, because images are already compressed by JBIG2, `pdfsizeopt.py` do not require any other types of compression methods.

Table 4 shows sizes of each collection before and after optimization. There are only counted PDF files of articles. The original articles contain compressed images by CCIT G4 as saved by FINEREADER. The second column shows size of each journal after running the PDF re-compressor. The last column shows the size after running both optimizers (PDF re-compressor and `pdfsizeopt.py`) and reached percentage of original size (to compare how much each collection's size was squeezed).

13. CONCLUSIONS AND FUTURE WORK

Automating the creation of useful digital libraries—that is, digital libraries affording searchable text and reusable output—is a complicated process, whether the original library is paper-based or already available in electronic form. (Simske and Lin [30])

In this paper, we have described the engineering aspects of the DML-CZ workflow, as decided, developed and tested during the project development. Most of the steps were carried out in our own laboratory, in the belief that we would gain expertize in and retain control over fine details. This would allow us to plug in new modules arising from leading edge research in the future—there are, currently, many new developments appearing and much research underway in the field of digitisation.

The complexity of the full digitisation workflow cannot be underestimated, especially when digitising heterogeneous sources. We have shown that after running both the PDF re-compressor and `pdfsizeopt.py` we were able to reduce the size of PDF documents to less than half. We can expect even better results when the planned improvements of the JBIG2ENC encoder are implemented.

We are now working on improving the modified JBIG2ENC and its comparison process in symbol coding by integrating some OCR techniques and tools. These techniques should also help us to decide which symbols should be taken as the representantive ones.

Instead of waiting for black-box solutions we decided to tackle the described issues by pilot studies and performed the actual digitisation task and necessary research. We believe that this is the most straightforward way towards the envisioned EUDML or even the World Digital Mathematics Library WDML. We foresee the methods, algorithms and tools developed impacting further afield than the EUDML.

Hundreds of decisions from very different areas have to be made, most of them crucial to the overall success of the project. Some subtasks may well be subcontracted, diminishing the expertize needed by the core project team members, and at the price of loosing flexibility and even quality due to a lesser holistic awareness of the project's intricacies. We have chosen to use open source software and approaches in the hope that they will be automated as much as possible.

The Internet connection and the developed software enable different specialized groups to work remotely. This approach lacks, however, the advantage of everyday personal contacts and discussions. It must be balanced with a thorough organization of the workflow, personal discipline, an effective quality control, and careful tracking of all parameters should problems arise.

Table 1: Average sizes (in bytes) of each type of PDF objects stored in single page, two-layer (bitmap+OCR below) PDF documents. PDF re-compressor uses modified JBIG2ENC with enabled symbol coding. `pdfsizeopt.py` uses MULTIVALENT and generic coding of JBIG2ENC, and does not use PNGOUT (has minimal effect if used or not because JBIG2 is the most common compression method used and not ZIP).

	Original PDF	After running PDF re-compressor	After using `pdfsizeopt.py`	After using both
Total size	135,870	105,119	70,957	63,428
Content objects	5,030	5,017	4,838	4,838
Font data objects	24,831	24,798	3,887	3,887
Header	16	15	15	15
Image objects	76,780	5,632	57,426	5,616
Linearized Xref table	787	0	0	0
Other objects	25,949	66,752	4,526	48,806
Separator data	20	22	22	22
Trailer	97	121	135	136
Wasted between objects	99	0	0	0
Xref table	2,259	2,762	108	108

Table 2: Average sizes (in bytes) of each type of PDF objects stored in *multi-page*, two-layer (bitmap+OCR below) PDF documents. All other is same as in Table 1.

	Original PDF	After running PDF re-compressor	After using `pdfsizeopt.py`	After using both
Total size (in kB)	7,123	4,702	3,962	2,717
Content objects (in kB)	308	308	297	297
Font data objects (in kB)	1,525	1,525	103	103
Header	15	15	15	15
Image objects (in kB)	4,717	1,915	3,529	1,904
Linearized Xref table	0	0	0	0
Other objects (in kB)	545	926	31	411
Separator data	124	24	24	24
Trailer	24	124	106	135
Wasted between objects	0	0	0	0
Xref table	28,188	28,208	1,361	1,200

Table 3: New size of PDF documents in comparison with the original ones (generated by FINEREADER and TEX).

	By using PDF recompressor		By using `pdfsizeopt.py`		By using both	
	single page	multi page	single page	multi page	single page	multi page
Saved globally	77.37%	66.01%	52.22%	55.62%	46.68%	**38.14%**
Saved in image and other objects	70.46%	53.99%	60.30%	67.66%	52.97%	44.00%

Table 4: Size of PDF documents from DML-CZ with bi-tonal images compressed originaly by CCIT G4 and replaced by their JBIG2 version and further optimized using `pdfsizeopt.py`. All values are in megabytes.

Journal or collection name	Size of original PDFs	Size after running PDF re-compressor	Size after running `pdfsizeopt.py`
Equadiff	279.45	194.27 (69.5%)	126.28 (45.1%)
NAFSA	79.50	59.19 (74.4%)	34.40 (42.1%)
Toposym	281.19	178.65 (63.5%)	144.80 (51.4%)
WSAA	469.60	300.23 (63.9%)	210.86 (44.9%)
WSGP	431.94	277.27 (64.1%)	183.09 (42.3%)
Časopis pro Pěst. Mat.	2,906.02	2,172.21 (74.7%)	1,296.12 (44.6%)
Časopis pro Pěst. Mat. Fys.	4,091.59	3,340.51 (81.6%)	1,700.13 (41.5%)
Czech Mathematical Journal	3,369.71	2,127.05 (63.1%)	1,874.04 (55.6%)
Kybernetika	2,297.92	1,646.02 (71.6%)	906.01 (39.4%)
Mathematica Bohemica	472.87	326.68 (69.0%)	234.18 (49.5%)
Mathematica Slovaca	2,725.65	1,895.06 (69.5%)	1,051.40 (38.5%)
Pokroky MFA	2,312.31	1,554.36 (67.2%)	858.44 (37.1%)
Bolzano Collection	534.05	348.47 (65.2%)	280.24 (52.4%)
Dějiny Mat.	170.45	115.71 (67.8%)	75.47 (44.2%)
Single books	170.60	117.07 (68.6%)	72.30 (42.3%)
Totals	**20,592.84**	**14,652.77 (71.1%)**	**9,047.77 (43.9%)**

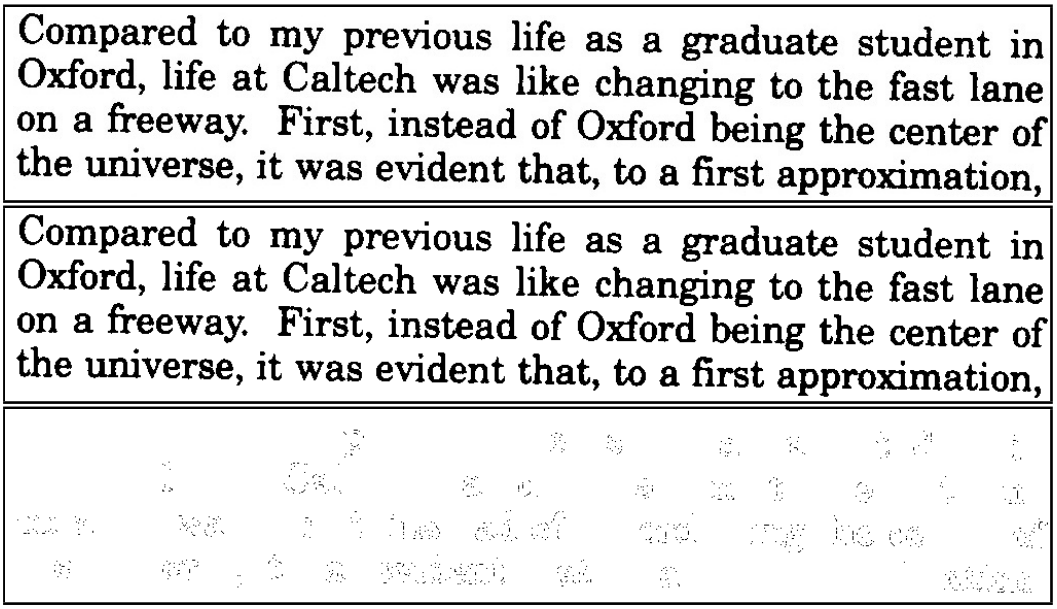

Figure 2: Example of part of the page before and after compression by JBIG2ENC. The lower image shows the differencies.

In the future we plan to:

- increase the coverage of DML-CZ's retro-born digital materials by developing parametrized semi-automatic conversions from PDF journal archives;
- finalize the metadata validation procedures;
- participate in defining the interfaces and conversion filters for data export for projects on European or worldwide levels;
- pursue research in the areas of mathematical document classification, math indexing and retrieval, mathematical expression OCR and document similarity [26] using data of collections DML-CZ and EUDML;
- improve PDF re-compressor by using INFTY and/or Tesseract for page/document subimages equivalence decisions and optimize it for speed;
- evaluate the possibility of assigning digital object identifiers (DOI) to papers in the DML-CZ, and to use the Handle System infrastructure `handle.net`;
- design alternative and novel user interfaces for the digital library; we are considering the graph spring layout (force-directed algorithms for graph drawing) and context techniques used in the TouchGraph project above the web of cross-referenced publications in the digital library (represented by configurable metadata views as in Visual Browser).

Acknowledgments

This research has been partially supported by the grant registration no. 1ET200190513 of the Academy of Sciences of the Czech Republic, EU project # 250,503 in CIP-ICT-PSP.2009.2.4, MŠMT CZ projects LC536 and 2C06009, Masaryk University grants for student research asistents #22,525/2010 and for support of high-quality creative activity of students no. LA09016. The authors thank other DML-CZ and EuDML colleagues for fruitful discussions and to the paper reviewers for improvement suggestions. Drawing of Figure 1 by Mirek Bartošek is acknowledged.

14. 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] M. Bartošek, P. Kovář, and M. Šárfy. DML-CZ Metadata Editor: Content Creation System for Digital Libraries. In Sojka [34], pages 139–151. `http://www.fi.muni.cz/~sojka/dml-2008-program.xhtml`.

[3] M. Bartošek and V. Krejčíř. Jak se dělá digitální matematická knihovna (in Czech). In *Proceedings of AKP 2007*, Liberec, Czech Republic, 2007. `http://dml.muni.cz/docs/akp2007-sbornik.pdf`.

[4] D. Bloomberg. Leptonica. [online], 2010. `http://www.leptonica.com/jbig2.html`.

[5] T. Bouche, T. Fischer, C. Goutorbe, and D. Ruddy. Digital Math Library Dublin Core (dml_dc): A Recommended Best Practice for Unqualified Dublin Core Metadata Records. Technical report, Cornell University, USA, Jan. 2008. `http://projecteuclid.org/collection/euclid/documents/metadata/dml_dc.html`.

[6] F. Brooks. The Computer Scientist as Toolsmith II. *Communications of the ACM*, 39(3):61–68, Mar. 1996.

[7] L. Bruno. IText PDF. [online], 2009. `http://www.itextpdf.com/`.

[8] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

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

[10] V. Dostál. Indexing of Mathematical Texts in the Digital Mathematics Library (in Czech). Master's thesis, Masaryk University, Brno, Faculty of Informatics, 2009.

[11] T. Dunning. Statistical identification of language. Technical Report MCCS 94-273, New Mexico State University, Computing Research Lab, 1994.

[12] E. S. Ferguson. *Engineering and the Mind's Eye*. MIT Press, Cambridge, MA, 1992.

[13] R. Hatlapatka. PDF Recompression using JBIG2. [online], 2010. http://nlp.fi.muni.cz/projekty/eudml/pdfRecompression/.

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

[15] A. Jackson. The Digital Mathematics Library. *Notices Am. Math. Soc.*, 50(4):918–923, 2003.

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

[17] V. Krejčíř. Building Czech Digital Mathematics Library upon DSpace System. In Sojka [34], pages 117–126. http://www.fi.muni.cz/~sojka/dml-2008-program.xhtml.

[18] L. Lalinský. Citation Crawling, 2009. Bachelor's Thesis Masaryk University, Brno, Faculty of Informatics, https://is.muni.cz/th/158017/fi_b/?lang=en.

[19] A. Langley. Homepage of jbig2enc encoder. [online]. http://github.com/agl/jbig2enc.

[20] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[21] I. Marosi and L. Tóth. OCR Voting Methods for Recognizing Low Contrast Printed Documents. In *Proceedings of Second International Conference on Document Image Analysis for Libraries (DIAL 2006)*, pages 108–115, Apr. 2006.

[22] T. Mudrák. Digitalizace matematických textů (in Czech, Digitisation of Mathematical Texts). Master's thesis, Masaryk University, Brno, Faculty of Informatics, Apr. 2006. https://is.muni.cz/th/60738/fi_m/?lang=en.

[23] R. Panák. Digitalizácia matematických textov (in Czech, Digitisation of Mathematical Texts). Master's thesis, Masaryk University, Brno, Faculty of Informatics, Apr. 2006. https://is.muni.cz/th/60587/fi_m/?lang=en.

[24] T. Pulkrábek. Obrazové transformace při digitalizaci textů (in Czech, Image Transformation during Digitisation), 2008. Bachelor's Thesis Masaryk University, Brno, Faculty of Informatics, https://is.muni.cz/th/139908/fi_b/?lang=en.

[25] R. Řehůřek and P. Sojka. Automated Classification and Categorization of Mathematical Knowledge. In S. Autexier, J. Campbell, J. Rubio, V. Sorge, M. Suzuki, and F. 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.

[26] R. Řehůřek and P. Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of LREC 2010 workshop New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, 2010. software available at http://nlp.fi.muni.cz/projekty/gensim.

[27] M. Růžička. Automated Processing of TEX-typeset Articles for a Digital Library. In Sojka [34], pages 167–176. http://www.fi.muni.cz/~sojka/dml-2008-program.xhtml.

[28] M. Růžička and P. Sojka. Data Enhancements in a Digital Mathematics Library. In Sojka [37], pages 69–76. http://dml.cz/dmlcz/702575.

[29] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24:513–523, 1988.

[30] S. J. Simske and X. 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.

[31] P. Sojka. From Scanned Image to Knowledge Sharing. In K. Tochtermann and H. 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.

[32] P. Sojka. Towards Digital Mathematical Library: Optical Character Recognition of Mathematical Texts. In J. Štuller and Z. Linková, editors, *Inteligentní modely, algoritmy a nástroje pro vytváření semantického webu*, pages 110–113, Prague, 2006. Ústav informatiky AV ČR.

[33] P. Sojka. Workflow in the digital mathematics library project: How mathematics is stored and retrieved. In J. Paralič, J. Dvorský, and M. Krátký, editors, *Proceedings of Znalosti 2006*, pages 243–247. VŠB–Technická univerzita Ostrava, 2006.

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

[35] P. Sojka. Digitization Workflow in the Czech Digital Mathematics Library. *Math-for-Industry Lecture Note Series*, 22:272–280, Dec. 2009.

[36] P. Sojka, editor. *Towards a Digital Mathematics Library*, Grand Bend, Ontario, CA, July 2009. Masaryk University. http://www.fi.muni.cz/~sojka/dml-2009-program.html.

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

[38] P. Sojka, R. Panák, and T. Mudrák. Optical Character Recognition of Mathematical Texts in the DML-CZ Project. Technical report, Masaryk University, Brno, Sept. 2006. presented at CMDE 2006 conference in Aveiro, Portugal.

[39] P. Sojka and R. Řehůřek. Classification of Multilingual Mathematical Papers in DML-CZ. In P. Sojka and A. Horák, editors, *Proceedings of Recent Advances in Slavonic Natural Language Processing—RASLAN 2007*, pages 89–96, Karlova Studánka, Czech Republic, Dec. 2007. Masaryk University.

[40] M. Suzuki, F. Tamari, R. Fukuda, S. Uchida, and T. Kanahori. INFTY — An integrated OCR system for mathematical documents. In C. Vanoirbeek, C. Roisin, and E. Munson, editors, *Proceedings of ACM Symposium on Document Engineering 2003*, pages 95–104, Grenoble, France, 2003. ACM.

[41] W. Sylwestrzak, J. Borbinha, T. Bouche, A. Nowiński, and P. Sojka. EuDML—Towards the European Digital Mathematics Library. In Sojka [37], pages 11–24. http://dml.cz/dmlcz/702569.

[42] P. Szabó. Optimizing PDF output size of TEX documents. *TUGboat*, 30(3):112–130, 2009. http://code.google.com/p/pdfsizeopt/.

[43] The Apache Software Foundation. Apache PDFBox – Java PDF Library. [online], 2010. http://pdfbox.apache.org/.