Mathematical Document Representation and Processing[1]

Petr Sojka

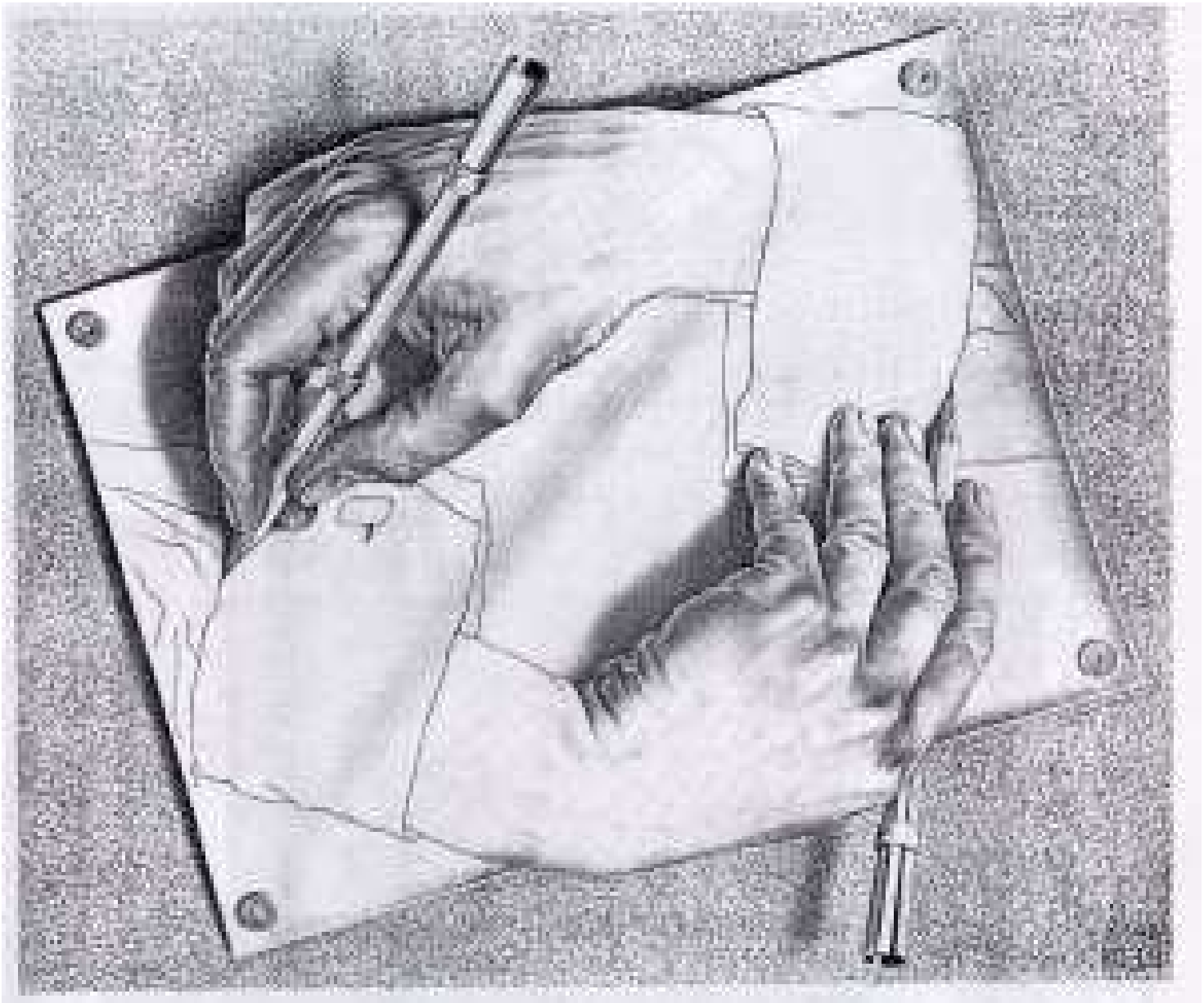
Faculty of Informatics, Masaryk University, Brno, CZ, EU

Dec 1st, 2008

[1] Supported by the Academy of Sciences of Czech Republic grant #1ET200190513

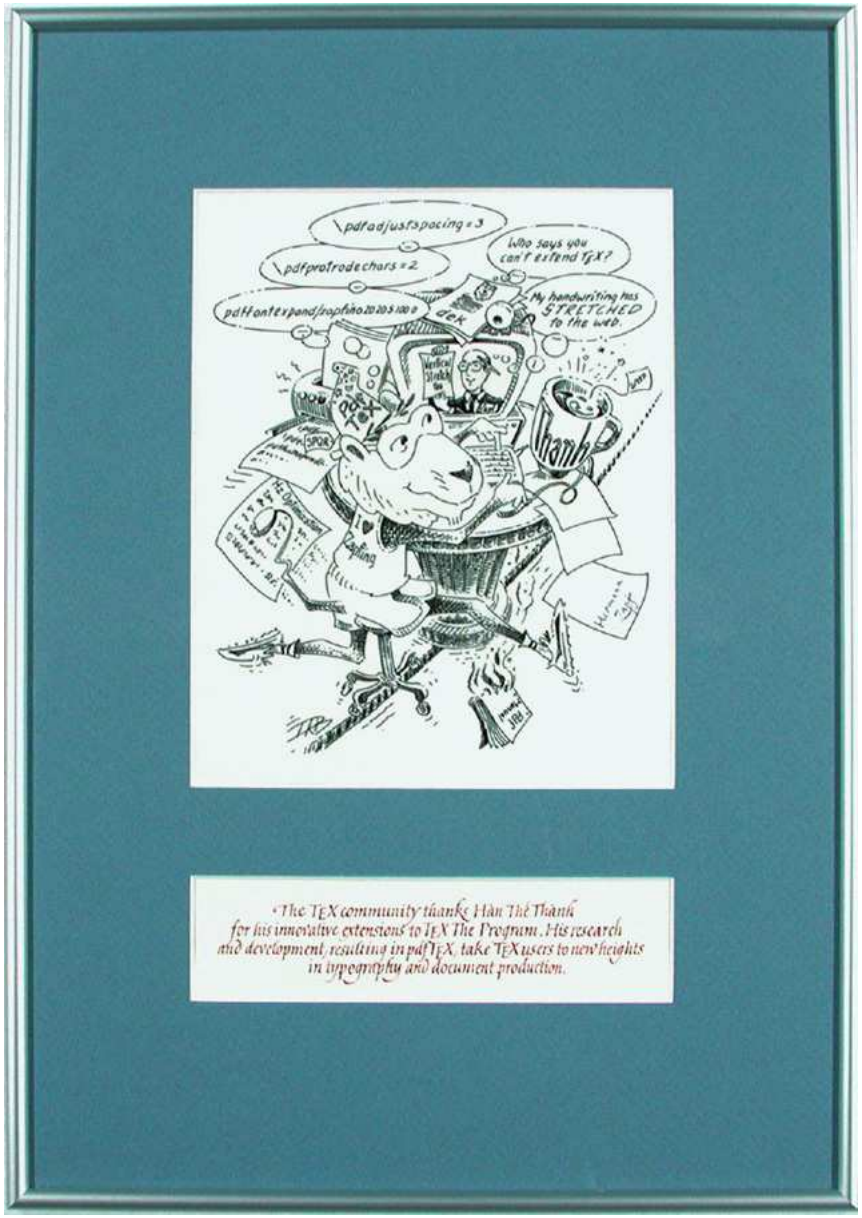
Purposes for storing mathematics

- Format choice **depends** on application's **purpose**.
- Most applications have its own internal format anyway.
- For exchange seems to win XML/MathML (but which one?).
- As authoring tools seems to be prefered (La)TEX.
- Quite different requirements have theorem proving systems.



Math Author Tools: LaTeX, AMSLaTeX (pdfAMSLaTeX)

- Good for authors: authors may express as close as possible to their mental model in their brain (new macros, namespaces).
- This author's advantage make headaches to the editors and those wishing to convert to some **standard** formalism (to index, evaluate, ...).
- Many different macropackages, and active development as possibilites grow (XeTeX, LuaTeX, pdfTeX), ...



pdfTEX program (used in Infty) was born in Brno during studies of Vietnamese student Hàn Thế Thành (1990–2001).

open-source, GPL/LPPL, new features added when needed as PDF format developes (every 18 month new PDF version), bugs fixed by pdfTEX team.

Widespread today, it in every Linux distribution, in given TeXlive 2008 DVD, ...

MathML: content vs. presentation

- MathML 2.0: XML namespace, W3C standard, supported and widely used.
- supported: in browsers (Firefox, IE, including fonts needed), symbolic computation sw (Mathematica, Maple), OCR sw (Infty :-)).
- de facto standard interapplication XML exchange format.
- extend to cover new things or not? (which DTD, symbol or notion e**X**tend/add?)

OpenMath and OMDoc

- **OpenMath**: markup language for specifying meaning of mathematical formula—complements MathML (used usually in presentation form only).
- Developed since 1993 in Europe (Helsinki), current president of OpenMath Society is Michael Kohlhase (MKM).
- For more richly structured content dictionaries (and generally for arbitrary mathematical documents) the **OMDoc format** extends OpenMath by a **statement level** (including structures like definitions, theorems, proofs and examples, as well as means for interrelating them) and a **theory level**, where a theory is a collection of several contextually related statements.
- Since 2000, now version 1.2, the richest of math formats today (exports easily to LaTeX or XHTML/MathML for presentation).

```
<OMOBJ xmlns=''http://www.openmath.org/OpenMath''>
  <OMA cdbase=''http://www.openmath.org/cd''>
    <OMS cd=''relation1'' name=''eq''/>
    <OMV name=''x''/>
    <OMA>
     <OMS cd=''arith1'' name=''divide''/>
      <OMA>
        <OMS cdgroup=''http://www.example.com/mathops'' cd=''multiops'' name=''
        <OMA>
          <OMS cd=''arith1'' name=''unary_minus''/>
          <OMV name=''b''/>
        </OMA>
        <OMA>
          <OMS cd=''arith1'' name=''root''/>
          <OMA>
            <OMS cd=''arith1'' name=''minus''/>
            <OMA>
              <OMS cd=''arith1'' name=''power''/>
              <OMV name=''b''/>
              <OMI>2</OMI>
            </OMA>
            <OMA>
              <OMS cd=''arith1'' name=''times''/>
              <OMI>4</OMI>
```

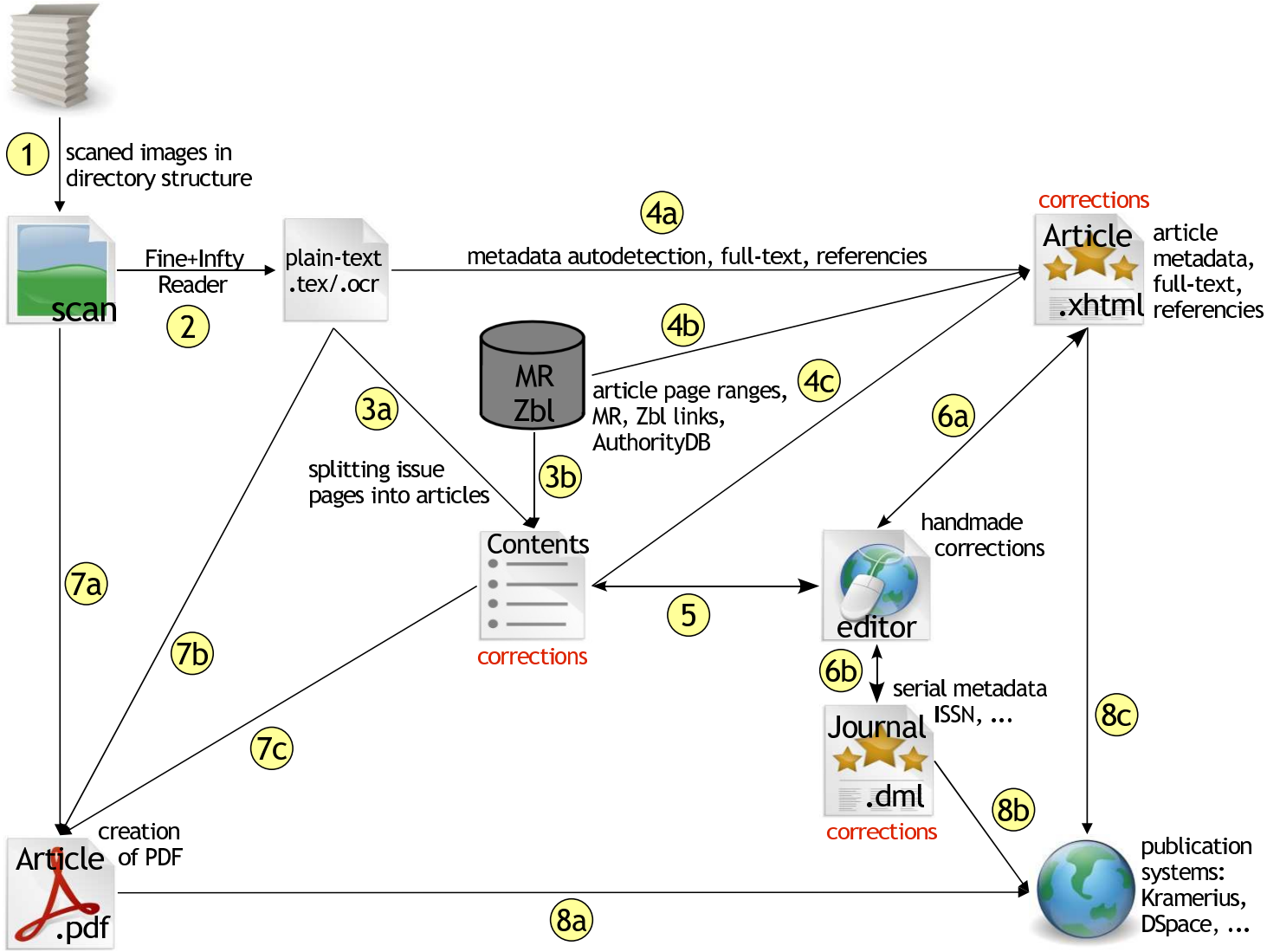
Semantically enhanced TEX—sTEX (by Michael Kohlhase)

- LATEX macropackage which will enable the author to add semantic information to the document in a way that does not change the visual appearance. This process is also referred to as semantic pre-loading and the collection of macro packages is called Semantic TeX (sTeX). Thus, sTeX can serve as a conceptual interface between the document author and MKM systems: Technically, the semantically pre-loaded LaTeX documents are transformed into the (usually XML-based) MKM representation formats, but conceptually, the ability to semantically annotate the source document is sufficient.
- To convey semantics to be convertible to OMDoc.
- Grabing most abstract semantic level, but not in widespread use by authors (additional effort does not pay back). Lack of motivation to be used.

Other formats

- DITA Darwin Information Typing Architecture: XML-based, end-to-end architecture for authoring, producing, and delivering technical information. This architecture consists of a set of design principles for creating "information-typed" modules at a topic level and for using that content in delivery modes such as online help and product support portals on the Web.
- OOXML OpenOffice XML (XML+ZIP).
- ODF OpenDocument Format (XML+ZIP).
- LaTeXML (Bruce Miller's mathematical encyclopaedia).

Top-level DML-CZ workflow (different primary data)



DML-CZ handling data from retro-digital period

- TIFF 4-bit ⟶ (geometrical corrections, binarization, noise filtering...) TIFF 1-bit ⟶ OCR.
- TIFF 1-bit ⟶ two-layer PDF.
- Metadata primary format (title, abstract, biblio): XML text UTF-8 + math in ASCII TEX ($...$).
- Before importing into digital library (DSpace in DML-CZ case) primary data are converted with pdfTEX into digitally signed PDFs with metadata in titlepage, bibtex file, and googlebot optimized web pages.
- Metadata available via OAI-PMH (maybe OAI-ORE in the future).

DML-CZ handling data from retro-born period

- only some source data and tools available now
- LaTeX, plainTeX, AMSTeX sources (heterogenity)
- automatic conversion possible rarely (conservative TeX modifications hard, as macros and font evolve)
- MathML and LaTeX conversions (TeX4ht, Tralics)
- Tralics for biblios (Archivum Mathematicum, Czech Mathematical Journal, Commentationes Mathematicae Universitatis Carolinae,...)

DML-CZ handling data born-digital period

- Different TEX flavours
- LATEX→ XML → normalized LATEX → PDF,... by River Valley Technologies for Springer and other publishing companies (usage of TeX4ht)
- Problems with new constructs when generating XML/MathML (continuous backward-compatible evolving)

Pilot project of Archivum Mathematicum

① inspired by CEDRAM

② papers in LaTeX with AMS styles, references in BibTeX.

③ new (modified) style files by Michal Růžička

④ automated typesetting, page numbering, EMIS web page generation,...

⑤ use of configurable Tralics converter to XML

⑥ high automation by program **make**

⑦ automated import to DML-CZ

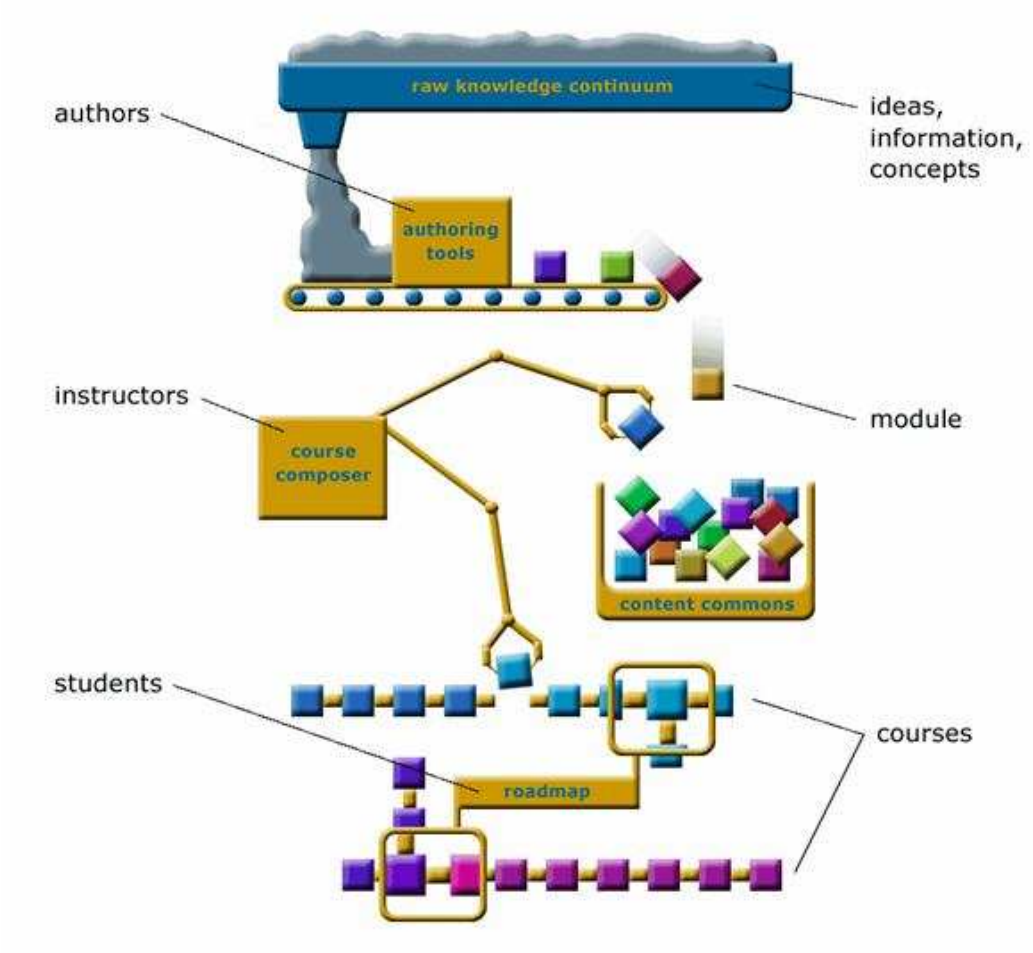
⑧ several issues already available in http://dml.cz

Teaching materials specifics: Connexions (Richard Baraniuk)

- many coauthors, change frequently
- pressure to cut down the textbook price
- electronic personalized version generated from primary, semantically rich sources (XHTML, PDF)
- Creative Commons publishing model

connexions

Connexions



Math indexing and search

- main text indexing tools (Lucene) using MathML structure tokens (big NSF grant, Robert Miner) **http://mathdex.com**
- indexing in EgoMath (Egothor v2) OSS search system (Galamboš, Mišutka): augmentation algorithm for formulae
- October 31, 2008 (Computerworld): *Google Inc. this week took another step in its effort to shed light on the so-called Dark Web, announcing that its engine can now search scanned documents in a PDF."*
  Math??

PDF

- file format by Adobe since 1993 to represent fixed layout 2-D (or 3-D objects in Acrobat 9 (Acrobat 3D, Acrobat Pro Extended))
- an open standard published on July 1, 2008 by the ISO as ISO 32000-1:2008.
- same imaging model as PostScript, but no scripting (viruses, slow) except possibly embedded JavaScript (ECMAScript).

PDF versions

- versions PDF 1.0 till 1.7 (every Acrobat came with new version)
- PDF/A optimized for archiving
- PDF/X* optimized for prepress world (portability)
- PDF ISO to meet wide goals of e-books
- digital signatures (PKI infrastructure), collaboration support
- Adobe applications support: HTML import (Web Capture), indexing, Adobe Access (TVR),...

JBIG2 (DJVU)

- image compression standard (from JPEG people) It compresses bitonal (black and white) only. Any "gray" areas must be simulated using black dots in a pattern called **halftoning**.
- parts: * Generic region coding * Symbol encoding (and text regions) * Refinement * Halftoning compressed with Huffman or arithmetic encoding

JBIG2's generic region encoding

- bitmap compression
- It is progressive and uses a context around the current pixel to be decoded to estimate the probability that the pixel will be black. If the probability is 50% it uses a single bit to encode that pixel. If the probability is 99% then it takes less than a bit to encode a black pixel, but more than a bit to encode a white one.
- The context can only refer to pixels above and to the left of the current pixel.

JBIG2's symbol encoding

- The idea of symbol encoding is to encode what a letter **A** looks like and, for all the **A**'s on the page, just give their locations (lossy encoding).
- However, assuming that we group the symbols correctly, we can get great compression this way. Remember that the stricter the classifier, the more symbol groups (classes) will be generated, leading to bigger files. But, also, there is a lower risk of cootoots (misclassification).

JBIG2's symbol retention

Symbol retention is the process of compressing multi-page documents by extracting the symbols from all the pages at once and classifing them all together. Thus we only have to encoding a single letter .a. for the whole document (in an ideal world).

This is obviously slower, but generates smaller files (about half the size on average, with a decent number of similar typeset pages).

One downside one should be aware of: If generating JBIG2 streams for inclusion to a linearised PDF file, the PDF reader has to download all the symbols before it can display the first page.

JBIG2's refinement

Symbol encoding is lossy because of noise, which is classified away and also because the symbol classifier is imperfect. Refinement allows us, when placing a symbol on the page, to encode the difference between the actual symbol at that location, and what the classifer told us was **close enough**. We can choose to do this for each symbol on the page, so we don't have to refine when we are only a couple of pixel off. If we refine whenever we find a wrong pixel, we have lossless encoding using symbols.

JBIG2's results: some numbers

Sample set of 90 pages scanned pages from the middle of a recent book. The scanned images are 300 DPI grayscale and they are being upsampled to 600 DPI 1-bpp for encoding.

- Generic encoding each page: 3435177 bytes
- Symbol encoding each page (default classifier settings): 1075185 bytes
- Symbol encoding with refinement for more than 10 incorrect pixels: 3382605 bytes

jbig2enc: OSS sw to generate JBIG2 files

- Free encoder **jbig2enc** developed by Adam Langley, in C: **http://www.imperialviolet.org/jbig2.html** supported by Google.
- Support for native embedding of JBIG2 files is in pdfTEX since 1.40.0-beta-20060811 (generated PDFs can be viewed by xpdf 3.01, Adobe Reader 5.0 upward, and by recent ghostscript versions)

pdfTEX JBIG2's Howto

Just create black and white thresholded .ppm files, and then you can compress multiple pages into one .jbig2 file with the jbig2 tool, e. g. by this Makefile entry:

```
jb2enc.jb2:     jb2enc1.ppm jb2enc2.ppm
        jbig2 -s jb2enc1.ppm jb2enc2.ppm > $@
```

then the pdfTeX primitives

```
\pdfximage page 1 {jb2enc.jb2} \pdfrefximage \pdflastximage
\pdfximage page 2 {jb2enc.jb2} \pdfrefximage \pdflastximage
```

should include both graphics. Similar with `\includegraphics` for pdfLATEX.

XHTML with MathML, DAISY, OMDoc

- XHTML with MathML portable, accessible, can be generated from LaTeX by TeX4ht (examples).
- DAISY for people with reading disabilities (text+audio).
- text-to-speech systems, TV Raman's AsTeR.
- OpenMath, OMDoc or sTeX as primary format for XHTML+MathML or DAISY generation.

Summary and conclusions

- No single all-purpose format winner, MathML close to it.
- XHTML+MathML for application exchange.
- PDF for final delivery to the masses, for bitmaps of scanned documents 600 DPI bitonal images should be saved with JBIG2 compression filter embedded in PDF since Acrobat 5.
- When conversions needed, a care has to be taken not to loose primary format qualities.
- Going as close to the author with the semantic tagging as possible, using machine learning, classification techniques for disambiguation, together with sets of heuristics. For born-digital, OMDoc or Content MathML are best.
- Will semantic web or Google indexing driven motivations help?

- Thank you for your attention! Question?