Extraction of Logical Structure from Articles in Mathematics *

Koji Nakagawa[1], Akihiro Nomura[2], and Masakazu Suzuki[1]

[1] Faculty of Mathematics, Kyushu University,
Kyushu Univ. 36, Fukuoka, 812-8581 Japan
{nakagawa,suzuki}@math.kyushu-u.ac.jp,
[2] Graduate School of Mathematics, Kyushu University

Abstract. We propose a mathematical knowledge browser which helps people to read mathematical documents. By the browser printed mathematical documents can be scanned and recognized by OCR (Optical Character Recognition). Then the meta-information (e.g. title, author) and the logical structure (e.g. section, theorem) of the documents are automatically extracted.
The purpose of this paper is to show the extraction method of logical structure specialized for mathematical documents. We implemented this method in INFTY which is an integrated OCR system for mathematical documents. In order to show the feasibility of the method we made a correct database from an existing mathematical OCR database, and made an experiment.

1 Introduction

Computers became indispensable devices for mathematics. This phenomenon can be seen by the success of mathematical systems (e.g. Mathematica or Maple) which have been being used for various other fields: physics, economics etc.

In order to apply mathematics to the real world, mathematical knowledge should be stored in computers in a way that people can easily use. Even if more and more mathematics is done in formal ways, most of mathematical knowledge is still stored in papers or books. Therefore digitizing mathematical text is still important.

1.1 Levels of Digitization

There are several kinds of mathematics digitization. In the paper [1], Adams gave some classifications of digitization of mathematics. Based on this consideration, in this paper we introduce five levels of mathematics digitization.

– level 1: bitmap images of printed materials (e.g. GIF, TIFF),

* This work is (partially) supported by Kyushu University 21st Century COE Program, Development of Dynamic Mathematics with High Functionality, of the Ministry of Education, Culture, Sports, Science and Technology of Japan.

- level 2: searchable digitized document (e.g. PS, PDF),
- level 3: logically structured document with links (e.g. HTML, MathML, LaTeX),
- level 4: partially executable document (e.g. Mathematica, Maple),
- level 5: formally presented document. (e.g. Mizar[6], OMDoc[4])

Currently most of mathematical knowledge is stored and used mainly in printed materials (level 1) like books or journals. For being used actively it is preferable that mathematical text is stored in a possibly higher level of digitization. However since making documents digitized to a higher level needs quite a lot of efforts, digitization of mathematical knowledge has not been enhanced so far. Therefore we definitely need software in order to automatize the digitization process in a possibly higher level.

1.2 Technologies for Automatization

The automatization can be achieved step by step:

- level 1 to level 2: OCR (Optical Character Recognition),
  In order to retrieve searchable digitized document from bitmap images, OCR is used. With OCR, character sequences can be recognized from bitmap images and then they can be used for searching words. Especially recognition of mathematical formulae is the most important in recognizing mathematical documents. The mathematical formulae recognition has been well investigated[8].
- level 2 to level 3: Extracting Logical Structure and Hyper Links,
  Obtained data after OCR are basically characters having positions in a page structured by lines and areas. They do not directly contain meta-information (e.g. author, title) of a paper and structural information (e.g. section, subsection, itemize). Also they do not have hyper links which point to internal and external documents.
- level 3 to level 4: Semantics Recognition from Presentation,
  Sometimes executable blocks (e.g. mathematical expressions, algorithms) appear in mathematical text. In level 3 mathematical expressions are described in the two-dimensional (presentational) way. We need to extract semantic expressions from these presentational expressions. Mathematica[10] has standard collections of these transformation rules which retrieve semantics of presentational expressions and one can even define their own style of notation (See `MakeExpression` function in [10]).
- level 4 to level 5: Understanding Mathematical Document,
  Usually mathematical statements like definitions, lemmata, theorems, proofs are written in natural languages in books or papers. Therefore for treating them in computers we need natural language processing. The first step of the natural language processing is parsing. For parsing it is common to make a corpus which is a set of grammar rules extracted from used expressions. Making a corpus for mathematical statements was done by Baba and Suzuki[2].

After parsing, formalizing written mathematical description to logical formulae in a predicate logic can be achieved. Formalized statements can be used for proving in computers by theorem provers like Theorema[3].

Since current our mathematical activities range over all digital levels, we need software which covers all aspects of these technologies from scanning to proving in a coherent manner. The ultimate goal is that scanned mathematical papers are processed and the software system gives us whether the proofs are correct, though this goal is very ambitious.

In Section 2 since the ultimate goal is quite ambitious, as a sub-goal we propose a mathematical knowledge browser which covers from level 1 to level 3. In order to make such a browser, some technologies are necessary. One of the technologies is to extract logical structure from documents after OCR. In Section 3 we discuss the method of extracting logical structure from mathematical documents. In order to show the feasibility of the method, we made a correct database which can represent logical structure information based on an existing mathematical database for OCR, and experimented for the correct database. In Section 4, the correct database is described and the result of the experiment will be shown. Then we conclude the discussion in Section 5.

2 Mathematical Knowledge Browser

The mathematical knowledge browser helps people to do mathematics from level 1 to level 3. One of inputs for this mathematical knowledge browser is a printed mathematical document. A printed document can be scanned, and then processed by OCR. After OCR logical structure and hyper links are automatically extracted and shown to users.

We will implement this mathematical knowledge browser on an integrated OCR system for mathematical documents called INFTY[8] (INFTY can be downloaded[3]). INFTY reads scanned page images of a mathematical document and provides their character recognition results. One of the distinguished characteristics of INFTY is that it can recognize two-dimensional mathematical expressions. INFTY has a graphical user interface which can show mathematical expressions in the ordinary two-dimensional mathematical notation, and has a manual error correction interface. The recognition results can be saved in various formats, e.g. XML (called KML), HTML, LaTeX, Mathematica, and braille.

2.1 User Interface

The browser consists of three panes: structure pane, reference pane, browsing pane (Fig. 1). In the structure pane located in the left side, structural information is shown as a tree like a file manager. The browser pane on the right bottom and the reference pane on the right top show mathematical text. By clicking a link in the browser pane, the text pointed by the link will be shown in the reference

[3] `http://infty.math.kyushu-u.ac.jp/index-e.html`

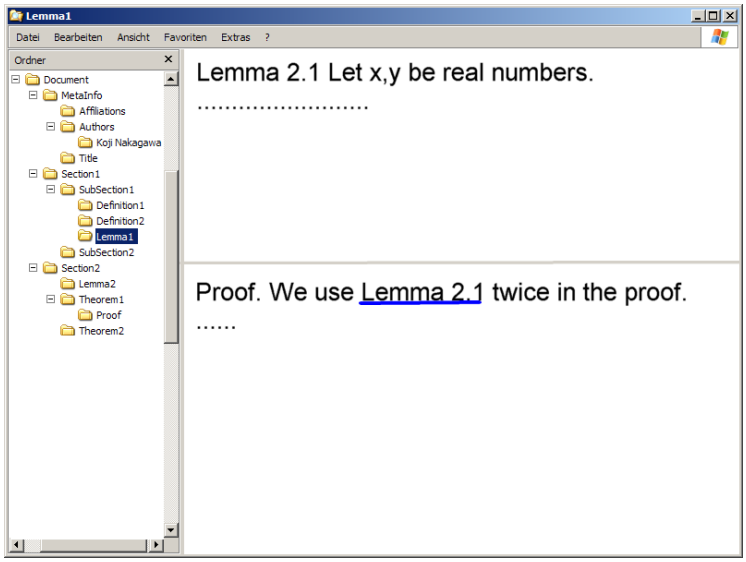


Fig. 1. Screen Image of Mathematical Knowledge Browser (Sketch)

pane so that people won't lose the attention in the browser pane. For example, by clicking a link 'Lemma 2.1' the reference pane shows 'Lemma 2.1' while the browser pane does not change.

2.2 Showing Relationship of Mathematical Theory Structure

Usually in a mathematical paper, mathematical components (e.g definitions, lemmata, theorems) have dependencies and one can construct a graph which shows the dependencies. With the mathematical knowledge browser one can see such dependency graphs. Fig. 2 shows an example of such a graph. For example, suppose 'Lemma 3.1' is used in the proof of 'Theorem 4.2', the text 'Lemma 3.1' should appear in the proof of 'Theorem 4.2'. From this fact we can detect the dependency automatically. By this functionality, readers can recognize theory structure of a paper before reading into details.

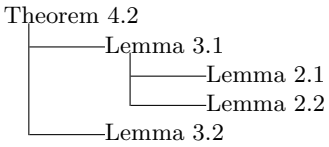


Fig. 2. Graph of Theory Dependency

Of course, showing theory structure is not a new idea. However the important point is that the theory structure can be automatically extracted from printed mathematical documents.

3 Extracting Meta-Information and Logical Structure

For realizing the mathematical knowledge browser, we need several technologies. One of the technologies is to extract meta-information and logical structure from mathematical documents. In this section, we discuss the method to extract automatically meta-information and logical structure.

There are several studies of logical structure extraction from documents. Extensive surveys can be found in papers[5, 7]. In these studies target documents vary from general documents to specific documents. The work presented in this paper is unique because it is specialized for mathematical documents and it extracts mathematical specific components (e.g. Theorem, Proposition). The method proposed in this paper does not need to know layout styles beforehand, while some other studies do.

3.1 Data Representation for Meta-Information and Logical Structure

INFTY[8] produces the following nested structure as output from scanned images as input.

- 1st page (size and position in the page)
  - 1st area (size and position in the page)
    - 1st line (size and position in the page)
      - 1st character (code, size and position in the page)
      - 2nd character (code, size and position in the page)
      - ...
    - 2nd line
    - ...
  - 2nd area
    - 1st line
      - 1st character
      - ...
    - ...
- 2nd page
- ...

A document contains several pages and each page contains several areas which have positions and sizes in the page. An area can contain lines which also have their positions and sizes. A line has recognized characters with positions, sizes and font styles.

INFTY produces the output in a XML format called KML. We extended the KML format so that it can represent meta-information and logical structure. For

example, Fig. 3 shows a result output in the extended KML for a scanned image shown in Fig.4. The top element is 'Doc' which contains some 'Sheet' elements representing pages. 'Sheet' elements contain some 'Area' elements whose positions and sizes are indicated by 'rect' attributes. The value of the 'rect' attribute "$left$,up,$right$,$down$" indicates the positions of left, up, right, down borders of the rectangle. An 'Area' element contains a 'Text' element having a 'Field' element. A 'Field' element has several 'Line' elements which have again several 'Char' elements.

For the need of putting additional information for meta-information and logical structure we added the 'tag' attribute for the 'Text' element in order to represent the type of the text field. The values of the 'tag' attribute are 'PageHeader', 'PageNumber', 'Caption', 'Title', 'AuthorInfo', 'AbstractHeader', 'Abstract', 'Keywords', 'Heading1', 'Heading2', 'Heading3', 'Heading4', 'Heading5', 'Text', 'ReferenceItem', 'Definition', 'Axiom', 'Theorem', 'MainTheorem', 'Proposition', 'Corollary', 'Lemma', and 'Footnote'.

3.2 Extraction Algorithm

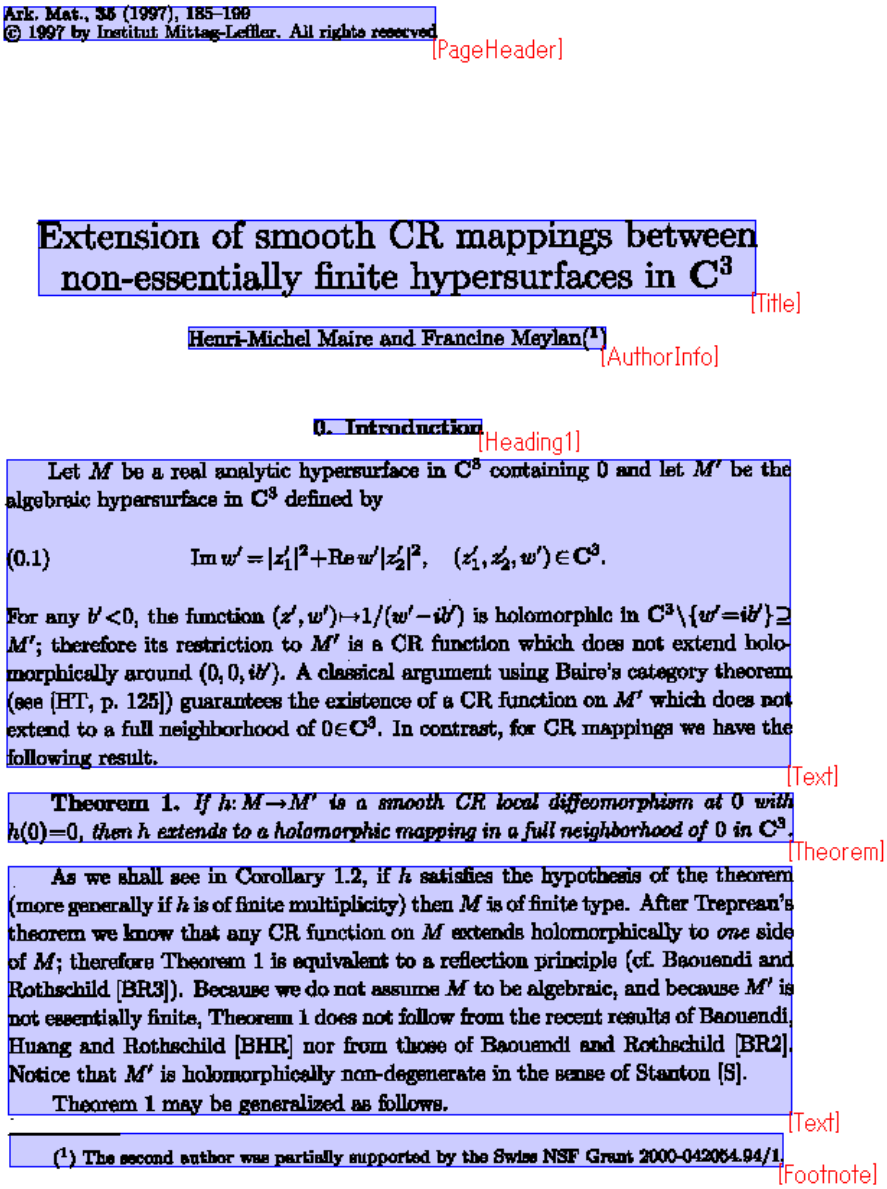
The algorithm of extracting logical structure works in two steps: segmenting areas in a page and putting appropriate tags to the areas. In Fig. 4, areas are indicated by gray rectangles and their tags are put beside the rectangles. An area can be either a special text area (page number, running header, captions of tables and figures, footnotes, and headings) or a normal text area which can be a mathematical component (e.g. theorem, definition). Later the method to extract mathematical components is described in details. After the correct process of putting tags, the conversion to a logically structured format (e.g. HTML, OMDoc) is straightforward.

Segmentation Segmentations can be done in the following three ways:

- Segmentation by Spacing
  By using spacing information, scanned images are separated into several areas which can be either text area, figure/table area, or formulae area.
- Segmentation by Style Difference
  For each text area, the average size of the characters contained in the area is calculated. The size of a character can be determined by the height of the character. Also boldness can be calculated by the horizontal widths of characters. In an area when the styles (bold, italic, and size) of lines are obviously different from those of other lines, they are separately segmented.
- Segmentation by Keywords
  In an area, when a special keyword of mathematical components (e.g. Theorem, Lemma, Definition) comes in the beginning of a line, basically they are segmented. However sometimes there are cases that keywords do not indicate beginnings of mathematical components. This issue will be discussed in the next subsection.

```
<Doc version="1.1" language="English" ...>
 <Sheet id="1" doc_file_name="Arkiv_1997.kml"
   image_file_name="Arkiv_1997_185.tif" height="4438" width="3015" ...>
  <Area rect="148,129,1801,266" id="1" ...>
   <Text rect="148,129,1801,266" tag="PageHeader" ...>
    <Field base_char_size="16,30,13,41" sub_char_size="11,20,9,28">
     <Line id="1" rect="148,129,1086,195">
      <Char code="0141" rect="148,133,195,180" ...>A</Char>
      <Char code="0172" rect="200,151,223,180" ...>r</Char>
      ...
     </Line>
     <Line id="2" rect="149,200,1801,266">
      ...
  </Area>
  <Area rect="279,948,3022,1239" id="2" ...>
   <Text rect="279,948,3022,1239" tag="Title" ...>
   ...
  </Area>
   ...
 </Sheet>
 <Sheet id="2" doc_file_name="Arkiv_1997.kml"
   image_file_name="Arkiv_1997_186.tif" height="4432" width="3002" ...>
  <Area rect="229,169,326,215" id="1">
   <Text rect="229,169,326,215" tag="PageNumber" ...>
    <Field base_char_size="16,28,12,39" sub_char_size="11,19,8,26">
     <Line id="1" rect="229,169,326,215">
   ...
   </Text>
  </Area>
  <Area rect="1088,168,2358,228" id="2">
   <Text rect="1088,168,2358,228" tag="PageHeader" ...>
    <Field base_char_size="16,29,12,40" sub_char_size="11,20,8,27">
     <Line id="1" rect="1088,168,2358,228">
  ...
  </Area>
  <Area rect="231,405,3224,701">
   <Text tag="Theorem">
    <Field>
     <Line id="1" rect="392,405,3224,486">
      <Char code="2154" rect="392,410,452,467" bold="1"...>T</Char>
      <Char code="2168" rect="458,409,506,467" bold="1"...>h</Char>
      ...
  </Area>
 <CharInfo>... </CharInfo>
</Doc>
```

Fig. 3. Example of KML Output from INFTY OCR Engine


Ark. Mat., 35 (1997), 185–199

[PageHeader]

Extension of smooth CR mappings between non-essentially finite hypersurfaces in $\mathbf{C}^3$

[Title]

Henri-Michel Maire and Francine Meylan([1])

[AuthorInfo]

0. Introduction

[Heading1]

Let M be a real analytic hypersurface in $\mathbf{C}^3$ containing 0 and let M' be the algebraic hypersurface in $\mathbf{C}^3$ defined by

$$\operatorname{Im} w' = |z_1'|^2 + \operatorname{Re} w' |z_2'|^2, \quad (z_1', z_2', w') \in \mathbf{C}^3. \tag{0.1}$$

For any $b'<0$, the function $(z', w') \mapsto 1/(w' - ib')$ is holomorphic in $\mathbf{C}^3 \setminus \{w' = ib'\} \supseteq M'$; therefore its restriction to M' is a CR function which does not extend holomorphically around $(0, 0, ib')$. A classical argument using Baire's category theorem (see [HT, p. 125]) guarantees the existence of a CR function on M' which does not extend to a full neighborhood of $0 \in \mathbf{C}^3$. In contrast, for CR mappings we have the following result.

[Text]

Theorem 1. *If $h: M \to M'$ is a smooth CR local diffeomorphism at 0 with $h(0)=0$, then h extends to a holomorphic mapping in a full neighborhood of 0 in $\mathbf{C}^3$.*

[Theorem]

As we shall see in Corollary 1.2, if h satisfies the hypothesis of the theorem (more generally if h is of finite multiplicity) then M is of finite type. After Treprean's theorem we know that any CR function on M extends holomorphically to *one* side of M; therefore Theorem 1 is equivalent to a reflection principle (cf. Baouendi and Rothschild [BR3]). Because we do not assume M to be algebraic, and because M' is not essentially finite, Theorem 1 does not follow from the recent results of Baouendi, Huang and Rothschild [BHR] nor from those of Baouendi and Rothschild [BR2]. Notice that M' is holomorphically non-degenerate in the sense of Stanton [S].

Theorem 1 may be generalized as follows.

[Text]

([1]) The second author was partially supported by the Swiss NSF Grant 2000-042064.94/1.

[Footnote]

Fig. 4. Example of Scanned Image Separated by Areas

Putting Tags to Areas After the segmentation process, appropriate tags are put to these segmented areas. Here we describe criteria to decide appropriate tags for areas.

- **Title and Headings** (e.g. section, subsection)
  In order to put tags to these area, areas are ordered by the following lexicographic ordering through a document.

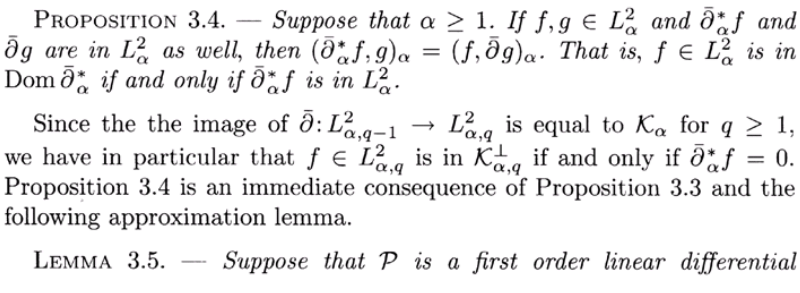
$$\langle Size, AllCapital, HCapital, Bold, Italic \rangle$$

$Size$:	average size of characters contained in the area
$AllCapital$:	true if all characters are written in capital (e.g. SYSTEM)
$HCapital$:	true if only head characters are written in capital (e.g. System)
$Bold$:	true if characters are written in bold
$Italic$:	true if characters are written in italic

  The 'Title' tag is put to areas appeared in the upper part of the first page and is written in the largest area in the paper according to the lexicographic order above. Headings usually start from either some numbers separated by periods or special keywords like 'Introduction', 'References', 'Bibliography' in an emphasized (large/bold/italic) font. The tags 'Heading1','Heading2', $\cdots$ are put to the second, third, $\cdots$ largest area in the lexicographic order.
- **Author Information**
  The tag 'AuthorInfo' is put to areas which come next to the title before 'Heading1'.
- **Page Header, Footnote, and Page Number**
  The 'PageHeader'('Footnote') tag is put to areas positioned in the top (bottom) of the page and written in smaller fonts than the average font size of the text areas in a paper. The 'PageNumber' tag is put to the areas which appear on the bottom or upper right or upper left of a page, and consist of numbers in Arabic style or in Roman style (e.g. i, ii, iii, and iv).
- **Mathematical Components** (e.g. definition, lemma, theorem)
  The tags for mathematical components are put to areas which start from the special keywords like 'Theorem', 'Definition', etc. Here the problem is that these special keywords do not always indicate beginnings of mathematical components. The problem will be discussed in the next subsection.

3.3 Two Difficulties in Extraction of Mathematical Components

Correct extraction of mathematical components are important for further processing of mathematical documents, e.g. indexing, construction of dependency graphs, and understanding of mathematical statements. Here we describe two difficulties for extracting mathematical components.



PROPOSITION 3.4. — *Suppose that* $\alpha \geq 1$. *If* $f, g \in L^2_\alpha$ *and* $\bar{\partial}^*_\alpha f$ *and* $\bar{\partial} g$ *are in* L^2_α *as well, then* $(\bar{\partial}^*_\alpha f, g)_\alpha = (f, \bar{\partial} g)_\alpha$. *That is,* $f \in L^2_\alpha$ *is in* $\mathrm{Dom}\, \bar{\partial}^*_\alpha$ *if and only if* $\bar{\partial}^*_\alpha f$ *is in* L^2_α.

Since the the image of $\bar{\partial} : L^2_{\alpha,q-1} \to L^2_{\alpha,q}$ is equal to $\mathcal{K}_\alpha$ for $q \geq 1$, we have in particular that $f \in L^2_{\alpha,q}$ is in $\mathcal{K}^\perp_{\alpha,q}$ if and only if $\bar{\partial}^*_\alpha f = 0$. Proposition 3.4 is an immediate consequence of Proposition 3.3 and the following approximation lemma.

LEMMA 3.5. — *Suppose that* $\mathcal{P}$ *is a first order linear differential*

Fig. 5. Looking for Beginnings of Mathematical Components

Looking for Beginnings of Mathematical Components Basic idea of detecting beginnings of mathematical components is to look for special keywords (e.g. Theorem, Lemma, and Definition). However these special keywords do not always indicate beginnings of mathematical components.

For example in Fig. 5, the 6th line starts with the keyword 'Proposition', but it is not the beginning of a proposition declaration. It appears in text in order to refer the 'PROPOSITION 3.4' defined above. In Fig. 5 proposition components start from the keyword with all capital characters 'PROPOSITION' which distinguish from other keywords like 'Proposition'. However we can not assume that keywords of beginnings are written in all capital characters, because there are many other papers which have different styles. Therefore we need a general algorithm in order to detect styles of mathematical components. In Section 3.4 the method will be described in details.

Looking for Endings of Mathematical Components Looking for the endings of the mathematical components is not so easy task. For example, in Fig. 6, the fifth line is the ending of the lemma. In this case, the lemma is written in italic till the fifth line and from the sixth line it turns into normal font. Therefore the ending of the lemma can be detected. It is usual that lemmata or theorems are written in italic. However definitions usually are not written in italic. Another criterion can be indentation or space between lines. In this paper, we simply look for the style change from italic to normal font for mathematical components except definitions, and for definitions we look for indentation.

When a mathematical component spreads over more than one page, failure detection of 'PageHeader', 'Footnote', 'PageNumber' causes failure of looking for the ending of the mathematical component. Therefore the detection of 'PageHeader', 'Footnote', 'PageNumber' is very important.

LEMMA 4.1. i) *If* $x \in H^*(\boldsymbol{G}/\boldsymbol{H}; k)$ *is transgressive with respect to the bottom fibering, then the element* $p^*(x) \in H^*(\boldsymbol{G}; k)$ *is universally transgressive.*

ii) *If* $x \in H^*(\boldsymbol{G}; k)$ *is universally transgressive then so is* $j^*(x) \in H^*(\boldsymbol{H}; k)$.

iii) *If* $H^i(\boldsymbol{G}/\boldsymbol{H}; k) = 0$ *for* $i < n$, $\deg x < n-1$ *for* $x \in H^*(\boldsymbol{G}; k)$ *and if* $j^*(x)$ *is universally transgressive, then* x *is also universally transgressive.*

These follow from the naturality of the transgression.

The following result is due to Borel [4] (see also Baum-Browder [2]).

LEMMA 4.2. *We can choose generators* $a, x_7, x_{11}, \cdots, x_{8n+8} \in H^*(\boldsymbol{PSp}(2n+1); \boldsymbol{Z}_2)$ *such that* $H^*(\boldsymbol{PSp}(2n+1); \boldsymbol{Z}_2) = \boldsymbol{Z}_2[a]/(a^4) \otimes \wedge(x_7, x_{11}, \cdots, x_{8n+8})$ *where* $\deg a = 1$ *and* $\pi^*(x_{4i-1}) = e_{4i-1}$, $i = 2, 3, \cdots, 2n+1$, *for the projection* $\pi : \boldsymbol{Sp}(2n+1) \to \boldsymbol{PSp}(2n+1)$.

Fig. 6. Looking for Endings of Mathematical Components

Line Feature	Explanation
BK-Definition	The line begins from the keyword 'Definition'.
BK-Theorem	The line begins from the keyword 'Theorem'.
BK-Proposition	The line begins from the keyword 'Proposition'.
BK-Corollary	The line begins from the keyword 'Corollary'.
BK-Lemma	The line begins from the keyword 'Lemma'.
K-Italic	The keyword is written in italic.
K-Bold	The keyword is written in bold.
K-LCapital	All characters of the keyword are in large capital. (e.g PROPOSITION)
K-SCapital	Most of characters of the keyword are in small capital. (e.g. PROPOSITION)
Indented	The line is indented.

Table 1. Line Features

3.4 Algorithm for Detecting Styles of Mathematical Components

In a paper, for a mathematical component the formatting style is uniform in principle. The idea of the algorithm for detecting styles of mathematical components is to use the style uniformity of a mathematical component.

The algorithm for detecting styles of mathematical components is described in Fig. 7. At first features shown in Table 1 are extracted from lines, and then for each line two style values 'lineDefStyle' and 'lineCompStyle' are caluculated. The variable 'lineDefStyle[i]' stores the style value of the ith-line and is for detecting the beginning of a definition mathematical component. The variable 'lineCompStyle[i]' is for detecting other mathematical components, since most of the cases the style of mathematical components except definitions is the same. If the line does not start from a special keyword of mathematical components, the style value becomes '-1' which means that it can not be the beginning of a mathematical component. Then the most frequent style values in a paper decide the styles of the beginning line of a mathematical component and the style values are assigned to the variables 'defStyle' and 'compStyle' which are the styles of mathematical components used in this paper. Finally when the style value of a line is not '-1' and it coincides with the most frequent value, the line is the

```
// calculating the style value for each line
for i=1 to line_length(paper) {
 lf=extract_features(paper.line[i]) // extracting line features
 // The number '-1' means that the line can not be
 // the beginning of a mathematical component.
 lineDefStyle[i]= if(lf.BK-Definition, stylefunc(lf), -1);
 lineCompStyle[i]=if(lf.BK-Theorem || lf.BK-Proposition ||
       lf.BK-Corollary || lf.BK-Lemma, stylefunc(lf), -1);
}
// detecting most frequent style values
defStyle =most_frequent(lineDefStyle);
compStyle=most_frequent(lineCompStyle);
// looking for endings
for i=1 to line_length(paper){
 if(!(compStyle==-1)&& lineCompStyle[i]==compStyle) {
         segmentArea();
         look_for_ending_by_Italic(i);}
 if(!(defStyle==-1) && lineDefStyle[i]==defStyle) {
         segmentArea();
         look_for_ending_by_Indent(i);}
...
};
int styleFunc(LineFeatures lf)
{ return (lf.K-Italic*2^0+lf.K-Bold*2^1+lf.K-LCapital*2^2+lf.K-SCapital*2^3
          +lf.Indented*2^4); }
 ...
 }
```

Fig. 7. Algorithm for Detecting Styles of Mathematical Components

beginning of a mathematical component and the process of looking for the ending is executed.

4 Experiment

In order to see the effectiveness of the algorithm we made an experiment for a database.

4.1 Outline of Database

We added new information of logical structure to a large-scale database of mathematical documents[9, 8] and made the correct database which can be used for the experiment. The documents contained in the database are 30 English articles on pure mathematics (issued in 1970 - 2000). Basically for each journal two old and new papers are selected. The numbers of pages and characters in the database are 422 and 706,279, respectively. This database is larger than other database used in the past researchers on math-OCR.

TagName	Correct	Success	Begin-AreaErr	End-AreaErr	TagErr	Miss-Recog.	Rate1 (%)	Rate2 (%)
PageHeader	467	447	0	11	9	0	95.7	95.7
PageNumber	439	418	0	11	10	0	95.2	95.2
Title	30	23	0	2	5	0	76.7	76.7
AuthorInfo	30	22	2	4	2	0	73.3	73.3
Abstract	5	0	0	2	3	1	0	20
Heading1	141	93	0	10	38	3	66	68.1
Heading2	25	8	4	6	7	0	32	32
Footnote	20	10	5	3	2	0	50	50
Definition	42	10	23	7	2	27	23.8	88.1
Theorem	130	99	17	12	2	20	76.2	91.5
Main Theorem	2	2	0	0	0	0	100	100
Proposition	96	73	10	11	2	14	76	90.6
Corollary	37	30	5	1	1	7	81.1	100
Lemma	116	89	11	16	0	18	76.7	92.2
Total	1580	1324	77	96	83	90	83.8	89.5

'Correct'	number of areas in the correct database.
'Success'	number of success tagging by the method proposed in this paper.
'Begin-AreaErr'	number of errors to detect beginnings of areas.
'End-AreaErr'	number of cases which detected beginnings of areas, but failed to detect endings of areas.
'TagErr'	number of errors that different tags were put.
'Miss-Recog.'	number of miss-recognitions by OCR for the special keywords.
'Rate1'	success rate computed by Success/Correct $*$ 100.
'Rate2'	success rate computed by (Success + Miss-Recog)/Correct $*$ 100.

Table 2. Experimental Result

All pages were scanned in 600 dpi and binarised automatically by the same commercial scanner (RICOH imagio Neo 450). The quality of resulting page images are noisy and include a lot of abnormal characters, such as touching characters and broken characters. The maximum and minimum abnormal character rates, which represent the quality of images, are 12.6% and 0.11%, respectively.

4.2 Experimental Result

We implemented the algorithm within the INFTY system and compared with the correct database described above. The result is summarized in the Table 2. If there are miss-recognition of characters, the keywords used for detection are not effective for extracting structure. Therefore it is considered that 'Success'+'Miss-Recog.' approximates the real success numbers for the method described in this paper.

(CONSIDERATIONS!!!) Increasing correct database Test for many papers!!!

5 Conclusion

A method of extracting meta-information and logical structure from mathematical documents was presented and implemented on the base of the INFTY system. In order to show the feasibility of the method, we made an experiment and got the result of ??% recognition rate.

The improvement of the recognition accuracy can be achieved by giving additional information to the system. For example, knowing journal names, which may be automatically extracted from running headers, can contribute to the accuracy, because a journal has its own format. Also when the system fails to detect logical structure, a little human interaction can contribute to the accuracy.

With the facility of automatic linking, we will be able to implement the mathematical knowledge browser based on the INFTY system. The mathematical knowledge browser can be extended to a system for editing mathematical documents. With the editor one can input mathematical statements in formalized formulae. The formalized formulae can be sent to computing, solving, proving services located in the Internet and retrieve the results. Namely it can be used as a front-end for mathematical services. We expect this mathematical knowledge browser to become a digitalization tool for mathematics, and enhance mathematical activities.

References

1. A. A. Adams. Digitisation, Representation and Formalisation: Digital Libraries of Mathematics. In *Mathematical Knowledge Management (MKM 2003), Feb. 16th - 18th, 2003, Bertinoro - Italy, LNCS 2594*. Springer, 2003.
2. Y. Baba and M. Suzuki. An Annotated Corpus and a Grammar Model of Theorem Description. In Andrea Asperti, Bruno Buchberger, and James Harold Davenport, editors, *Second International Conference, MKM 2003, Bertinoro, Italy*, pages 93–104, Feb. 2003.
3. B. Buchberger, C. Dupre, T. Jebelean, F. Kriftner, K. Nakagawa, D. Vasaru, and W. Windsteiger. The Theorema Project: A Progress Report. In *Symbolic Computation and Automated Reasoning*, pages 98–113. A.K. Peters, 2001.
4. M. Kohlhase. OMDoc: An Infrastructure for OpenMath Content Dictionary Information. *SIGSAM Bulletin (ACM Special Interest Group on Symbolic and Algebraic Manipulation)*, 34(2):43–48, 2000.
5. S. Mao, A. Rosenfeld, and T. Kanungo. Document Structure Analysis Algorithms: A Literature Survey. In *Document Recognition and Retrieval X*, number 5010 in Proceedings of SPIE, pages 197–207, January 2003.
6. P. Rudnicki. An Overview of the Mizar Project. In *Proceedings of the 1992 Workshop on Types for Proofs and Programs, Chalmers University of Technology, Bastad*, 1992. http://mizar.org.
7. K. M. Summers. *Automatic Discovery of Logical Document Structure*. PhD thesis, Cornell University, 1998.
8. M. Suzuki, F. Tamari, R. Fukuda, S. Uchida, and T. Kanahori. INFTY — An Integrated OCR System for Mathematical Documents. In *ACM Symposium on Document Engineering (DocEng '03), Grenoble, France, Nov. 20-22*, 2003.

9. S. Uchida, A. Nomura, and M. Suzuki. Quantitative Analysis of Mathematical Documents. Submitted to a journal.
10. S. Wolfram. *The Mathematica Book, Fifth Edition.* Wolfram Media, Inc., 2003.