An Annotated Corpus and a Grammar Model of Theorem Description

Yusuke Baba[1] and Masakazu Suzuki[2]

[1] Graduate School of Mathematics, Kyushu University
[2] Faculty of Mathematics, Kyushu University
6-10-1 Hakozaki, Higashi-ku, Fukuoka, 812-8581 Japan
{ma201040,suzuki}@math.kyushu-u.ac.jp

Abstract. Digitizing documents is becoming increasingly popular in various fields, and training computers to understand the contents of digitized documents is of growing interest. Since the early 90's, research of natural language processing using large annotated corpora such as the Penn TreeBank has developed. Applying the methods of corpus-based research, we built a syntactically annotated corpus of theorem descriptions, using a book of set theory, and extracted a grammar model of theorems from the obtained corpus, as the first step to understanding mathematical documents by computer.

1 Introduction

In recent years, digitizing documents has become increasingly popular in various fields, for example, in mathematics[1][2][3][4]. In connection with this movement, understanding the contents of digitized documents by computer is of growing interest[5][6].

The technology of understanding documents is applicable to useful systems such as machine translation, summarization and search. In mathematics, there are also particular application ideas of application e.g. the a system that translates descriptions of a proof into a language of proof-checker.

The fundamental technology of understanding documents by computer is parsing. The effectiveness of the above-mentioned systems depends heavily on the accuracy of the parser. Generally, parsing is complicated by the characteristics of natural language such as ambiguity, omission and inversion. On the other hand, since mathematical documents are written in logical and precise expressions, we can expect that the typical methods of parsing for natural language give more accurate results for mathematical documents.

The availability of large, syntactically annotated corpora such as the University of Pennsylvania Tree Bank(Penn TreeBank,[7]) lead to rapid developments in the field of natural language processing. Sekine et al.[8] extracted rules of grammar from the Penn TreeBank and released a parser of English, the "Apple Pie Parser[9]", using the grammar.

To extract a grammar from the Penn TreeBank, the first approach of Sekine et al. is based on the assumption that the corpus covers most of the possible

sentence structures in a domain. The outline of their idea follows. They assign parts-of-speech to an input sentence using a tagger, and then simply search for the same sequence of parts-of-speech in the corpus. The structure of the matched sequence is the output of their parser.

However, it turned out that this strategy was not practical. Only 4.7% of sentences in the corpus had the same structure as another sentence in the corpus. As a result, they applied the idea to not only sentences but also noun phrases, and extracted rules of grammar about S(sentence) and NP(noun phrase). 77.2% and 98.1% of S and NP structures have the same structure in the corpus, respectively [8].

If there were a large corpus covering most of the possible sentence structures in a domain, the method [8] would be more effective. While building a large corpus of natural language costs a great deal of labor and time, a corpus of theorem descriptions covering all the possible structures looks buildable with comparative ease because theorems have a limited vocabulary and consist of many idiomatic expressions. For this reason, we apply the corpus-based methods to process theorem descriptions.

We built a simply annotated corpus of theorem descriptions using a book of set theory, and extracted a context-free grammar with only three non-terminals from the corpus. The constructed corpus included about 100 instances, and the grammar that was extracted from the corpus had 141 generation rules. In order to evaluate the descriptive power of the obtained grammar, we performed an experiment as described in section 4.

2 Building a Corpus

2.1 Preliminaries

We used a book of set theory[10] to collect samples of theorem descriptions as the source of the corpus.

To build an annotated corpus of theorems, we used 28 categories of Parts-Of-Speech(POS) symbols, and 3 categories of phrase and clause symbols.

We processed the theorem descriptions as a sequence of words. In our corpus, a formula is simply a word.

2.2 Sentence structure

To express the structure of theorem descriptions in the corpus, we defined a symbol of phrase(**NP**) and two symbols of clause(**S, IFC**), as follows.

Proposition-Clause(S)
S is a string of words which expresses a proposition. **S** can include any other propositions in itself. Naturally, a theorem description is described by S.

Examples of S

- If R is a strict order on X , then S is a non-strict order on X.
- $(x_1, x_2) = (y_1, y_2)$ if and only if $x_1 = x_2$ and $y_1 = y_2$.
- X is an open set
- $V_\alpha \subset V_\beta$

If-Clause(IFC)
IFC is a string of words which expresses the assumption of a proposition.

Examples of IFC

- **If** $|\mathbf{X}| \leq |\mathbf{Y}|$ **and** $|\mathbf{Y}| \leq |\mathbf{X}|$, then $|X| = |Y|$.
- **In ZFC**, every vector space has a basis.

Noun-Phrase(NP)
NP is a string of words which can be processed like a noun.

Examples of NP

- "any non-zero ordinal"
- "the set $P = \{R(x) : x \in X\}$ of R-classes"
- "$P = \{R(x) : x \in X\}$"
- "X"

2.3 Classification of Parts-Of-Speech

Referring to the Penn TreeBank which has 48 categories of POS symbols[7], we use 7 original categories and 21 typical categories of POS on the grounds that the target domain of our corpus is only theorems. We defined original categories and typical categories of POS so that all the words that appeared in the samples would be classified naturally.

(a) Original Parts-Of-Speech
Theorem descriptions have many idiomatic expressions. For example, out of the 96 instances in the samples, 50 theorems(52.1%) have the structure:
"**If** [*proposition*], **then** [*proposition*]" or "**Let** [*proposition*], **then** [*proposition*]."
11 theorems(11.5%) have the structure: "[*proposition*] **if and only if** [*proposition*]." The words 'if' and 'let' are generally classified as a conjunction and a verb respectively, and 'if and only if' is generally partitioned into 'if' 'and' 'only' 'if', in natural language processing. However, since these words have particular meaning in theorem descriptions, they should be given special treatment. In this way, we defined 7 original categories of POS referring to empirical knowledge of mathematics and rough characteristics obtained from the samples:
(1) Proposition-Conjunction(PPC),
(2) Assumption(IF),

(3) Existence(EX),
(4) Restriction(RST),
(5) Number(NUM),
(6) Explain-Conjunction(EXPC),
(7) Formula(FML).
We give detailed explanations for these categories below.

(1) Proposition-Conjunction(PPC)

PPC is used between two propositions, and expresses a relationship between the propositions.

Examples of PPC

- if and only if
- implies

Examples of Appearances in Samples

- A set is most countable **if and only if** it is finite or countable.
- $P(\alpha)$ **implies** $P(s(\alpha))$ for any cardinal α.

(2) Assumption(IF)

IF is a word used in a description which expresses an assumption of the theorem (or a larger proposition which includes it).

Examples of IF

- if
- let
- suppose that
- assume that

Examples of Appearances in Samples

- **If** P is a partition of X, then $R = \{(x, y) : x, y \in p \text{ for some } p \in P\}$ is an equivalence relation on X.
- **Let** f be a function from X to Y. Then $\mathrm{Im}(f)$ is a subset of Y.

(3) Existence(EX)

EX is used to express the existence of something in a mathematical sense.

Examples of EX

- there is
- there exists

Examples of Appearances in Samples

- **there is** a bijective function from X to Y
- **there exists** a positive integer n

(4) Restriction(RST)

RST is used in order to impose restrictions on **NP**, in a mathematical sense.

Examples of RST

- such that
- satisfying

Examples of Appearances in Samples

- There is no set S **such that** $x \in S$ if and only if $x \notin x$.
- there is a set $Z \in X$ **such that** $p(Z) = Z$

(5) Number(NUM)

In mathematics there are various quantities, not only one, two, three, $\cdots$, but also 'only one', 'all', 'infinite' $\cdots$, which have particular meanings. **NUM** expresses such a numerical quantity in a mathematical sense.

Examples of NUM

- a
- unique
- any
- only one
- no

Examples of Appearances in Samples

- **a** well-ordered set
- For **any** formula ϕ, $\cdots$
- There is **no** set S such that $x \in S$ if and only if $x \notin x$.

(6) Explain-Conjunction(EXPC)

EXPC is used between propositions, and expresses an explanation or a translation.

Examples of EXPC

- in other words
- that is

Example of Appearances in Samples

- For any set X, there is an injection from X to PX but no bijection between these sets; **that is**, $|X| < |PX|$.

(7) Formula(FML)

FML expresses a mathematical formula. We process a formula as a word.

Examples of FML

- $p : PX \to PX$
- $f[A] = \{f(a) : a \in A\}$
- $(x_1, x_2) = (y_1, y_2)$
- X
- n

(b) Typical Parts-Of-Speech

In addition to the Original POS, we defined 21 categories of POS that are generally used in natural language processing. **Table 1** is a list of these Typical POS.

Using **S**, **IFC**, **NP** and POS, we assigned the sentence structure for each member of the samples. The structure of a sentence can be expressed as a tree. An example of the expression of a sentence structure is shown in **Fig. 1**. In the Figure, a box contains those parts of a sentence processed as a word. `IF, EX,` $\cdots$ are POS. Symbols of phrase and clause are assigned to nodes of the structure tree.

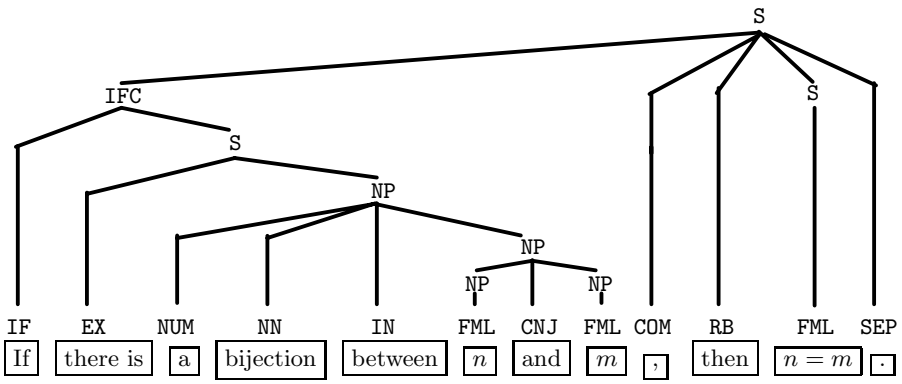

Fig. 1. Example of expression of sentence structure

Table 1. Typical POS

Symbol	Category	Examples of members
NN	Noun	set, relation, function, $\cdots$
PR	Pronoun	it, we, following, $\cdots$
RPR	Relative Pronoun	what, which, that, whose, $\cdots$
JJ	Adjective	countable, empty, same, $\cdots$
BE	be	is, are, be, $\cdots$
VB	Verb	have, lie, $\cdots$
VBG	Verb, gerund/present participle	subsutituting, asserting, $\cdots$
VBN	Verb, past participle	based, denoted, $\cdots$
MD	Modal	may, can, $\cdots$
RB	Adverb	mutually, moreover, totally, $\cdots$
CNJ	Conjunction	and, but, or, as, $\cdots$
IN	Preposition	on, of, in, from, to, $\cdots$
DT	Determiner	the, $\cdots$
LPAR	Left bracket character	(, {, [
RPAR	Right bracket character	), },]
LOQ	Left open quote	‘ *-and-* “
RCQ	Right close quote	’ *-and-* ”
SFP	Sentence-final punctuation	.
COM	Comma	,
COL	Colon, Semi-colon	: *-and-* ;
SYM	Other Symbols	- , * , $\cdots$

Our primary corpus has 96 instances of theorem descriptions which are all the theorems in [10]. **Table 2** shows some examples of members of the corpus. An instance of the corpus has 4 categories of data (a,b,c,d in the Table). Practically, we built the corpus in an XML file using tag hierarchy.

Table 2. Examples of Members of the Corpus

a: Character String of the Theorem Description
b: Partition of String into Words
c: Sequence of POS
d: Structure of Sentence
(the numbers 'n' appearing in the expression
represent the 'n-th' member of a sequence of POS)

a	$(x_1, x_2) = (y_1, y_2)$ if and only if $x_1 = x_2$ and $y_1 = y_2$.
b	$[(x_1, x_2) = (y_1, y_2)]$ [if and only if] $[x_1 = x_2]$ [and] $[y_1 = y_2]$ [.]
c	FML PPC FML CNJ FML SFP
d	(S (S (S 1) 2 (S (S 3) 4 (S 5))) 6)

a	There is no set S such that $x \in S$ if and only if $x \notin x$.
b	[There is] [no] [set] $[S]$ [such that] $[x \in S]$ [if and only if] $[x \notin x]$ [.]
c	EX NUM NN FML RST FML PPC FML SFP
d	(S (S 1 (NP 2 3 (NP 4) 5 (S (S 6) 7 (S 8)))) 9)

a	The union of at most countably many at most countable sets is at most countable.
b	[The] [union] [of] [at most] [countably] [many] [at most] [countable] [sets] [is] [at most] [countable] [.]
c	DT NN IN RB RB JJ RB JJ NN BE RB JJ SFP
d	(S (S (NP 1 2 3 (NP 4 5 6 (NP 7 8 9))) 10 11 12)13)

a	If there is an injective function from X to Y , and $X \neq \phi$, then there is a surjective function from Y to X.
b	[If] [there is] [an] [injective] [function] [from] $[X]$ [to] $[Y]$ [,] [and] $[X \neq \phi]$ [,] [then] [there is] [a] [surjective] [function] [from] $[Y]$ [to] $[X]$ [.]
c	IF EX NUM JJ NN IN FML IN FML COM CNJ FML COM RB EX NUM JJ NN IN FML IN FML SFP
d	(S (IFC 1(S 2(NP 3 4 5 6 (NP 7) 8 (NP 9))10 11 (S 12)) 13 14(S 15(NP16 17 18 19 (NP 20) 21 (NP 22)))23)

3 Extraction of Grammar

3.1 Algorithm

The structure tree of the corpus that we built has three non-terminal symbols : `S, IFC, NP`. We applied the method [8] to the corpus and three symbols, and extracted generation rules of a grammar. The algorithm for extracting the rules is as follows. Let G be the set of rules of the grammar to be extracted.
Repeat step1. and step2. for all the trees of the corpus.

The Algorithm of Extraction of Rules

Let G be the set of rules of the grammar to be extracted.
Repeat step1. and step2. for all the trees of the corpus.

step 1.
For the root and all the nodes of the tree,
make a generation rule of the grammar such that
the Left-hand-side of the rule is
the symbol of the node or root, and
the Right-hand-side of the rule is
the left-to-right sequence of the symbols of the children of the node or root.

step 2.
Add the rules made in step1. to G
unless any rule is already in G.

Ex. The Rules obtained from **Fig. 1** are as follows.

- `S -> IFC COM RB S SFP`
- `S -> EX NP`
- `S -> FML`
- `IFC -> IF S`
- `NP -> NUM NN IN NP`
- `NP -> NP CNJ NP`
- `NP -> FML`

3.2 The Obtained Grammar

Using Algorithm 3.1, we extracted a grammar which has 141 generation rules, as the model of theorem description. The obtained rules consist of 48 `S`-rules, 17 `IFC`-rules, and 76 `NP`-rules. **Table 3** shows a selection of the obtained rules. Since the grammar has only 3 non-terminals, some rules have a long right-hand-side.

Table 3. A selection of the obtained rules of grammar

S-rules (Total 48)
S -> EX NP
S -> FML
S -> IFC SFP RB S
S -> IN NUM NP COM S
S -> NP BE JJ
S -> NP BE NP
S -> NP MD RB VB NP
S -> NP VB
S -> PR BE JJ CNJ JJ
S -> PR BE NP LPAR PR BE COM JJ RPAR
S -> S CNJ S
S -> S COL EXPC COM NP S
...

IFC-rules (Total 17)
IFC -> IF S
IFC -> IF S COM IF S
IFC -> IF S SFP COM IF S
IFC -> IN NP
IFC -> IN NUM NP COM IF S
...

N-rules (Total 76)
NP -> DT JJ NN
NP -> DT NN IN JJ NN
NP -> DT NN IN NN IN NP IN NP
NP -> DT NN IN NN IN NP RPR VB IN NUM DT NN NP
NP -> NN NN
NP -> NP CNJ NP
NP -> NP CNJ PR NP
NP -> NUM JJ CNJ JJ NN
NP -> NUM JJ JJ NN IN NN RB NP
NP -> NUM JJ NN
NP -> RB JJ NN
NP -> RB RB JJ NP
...

4 Experiment

To evaluate the descriptive power of the grammar obtained in section 3(let G be the grammar), we used 100 theorems collected from two books A[11] and B[12] which are written about Galois theory and model theory.

We assigned the correct structure of **S**, **IFC**, **NP** and POS to each theorem, and extracted generation rules using algorithm sec:al. If all the rules obtained from a theorem are elements of G, then the theorem can be described by G. We checked whether or not each theorem is described by G. Additionally, we expanded G into G' by adding the rules which are not in G whenever they appear, and we then checked the descriptivity of G'.

Firstly, we experimented using 50 theorems of book A(Step-1). Secondly we used 50 theorems of book B and G' obtained from Step-1(Step-2). **Table 4** shows the results of the experiments. "G-describable" means that the theorem can be described by G. From the table, we can see that G describes only 10% of new theorems. Using the expanded grammar G' did not produce a significant increase in descriptive power.

We can consider that one factor influencing the results is a simplistic structure model. Because our structure has only 3 non-terminals, obtained rules have greater diversity in their right-hand-sides. Moreover, the fact that the total number of rules of G' increased at a constant rate means that the test corpus is too small.

Therefore in order to improve the system, we have to introduce other structures like verb-phrase or adjective-phrase, and expand our corpus.

Table 4. Experimental Results

	G-describable	G'-describable	Total number of rules of G' after Step
Step-1	5/50	11/50	223
Step-2	5/50	11/50	316

5 The Corpus Operation System

To build a large, syntactically-annotated corpus of mathematical text is troublesome work, and very time consuming. In order to make the work easier, we are designing a system that assists with many operations regarding using the corpus.

Fig. 2 is an image of the system. Users need only assign the correct structure of a sentence via a user-interface, and they can see the sentence structure being edited in real time as a tree. Other operations: tagging, parsing, expanding the corpus, training the grammar, are all automatic.

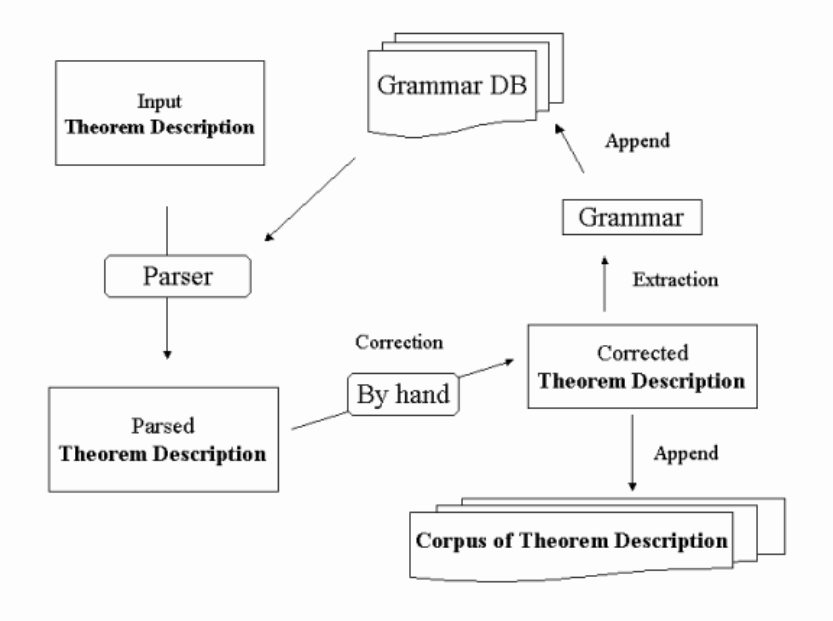

Fig. 2. The Corpus Operatrion System

6 Conclusion

We collected about 100 theorem descriptions from a book of set theory[10], and built a model for an annotated corpus of theorems.

We extracted a grammar model of theorem description from the corpus, which has 141 generation rules. We also performed an experiment to evaluate the descriptive power of the grammar.

The results of the experiment indicated that the structure of our test corpus is too simplistic to generate an effective grammar, and the size of the corpus is too small.

As the next step in our work, we will introduce another structure and collect more instances for the corpus. Additionally, we plan to build the semi-automatic corpus operation system in the near future.

References

1. Inoue, K., Miyazaki, R., Suzuki, M.: Optical Recognition of Printed Mathematical Documents. Proceedings of the Third Asian Technology Conference in Mathematics. Springer-Verlag. (1998) 280-289
2. Eto, Y., Suzuki, M.: Mathematical Formula Recognition Using Virtual Link Network. Proceedings of the Sixth International Conference on Document Analysis and Recognition. Seattle. IEEE Computer Society Press. (2001) 430-437

3. Michler, G.: A prototype of a combined digital and retrodigitaized searchable mathematical journal. Lecture Notes in Control and Infomation Sciences. **249** (1999) 219-235
4. Michler, G.: Report on the retrodigitiization project "Archiv der Mathemark". Archiv der Mathemark. **77** (2001) 116-128
5. IPSJ Magazine. vol.41 No.7 July (2000) (in Japanese)
6. IPSJ Magazine. vol.41 No.11 Nov. (2000) (in Japanese)
7. Marcus, M., et al.: Building a large annotated corpus of English:the Penn TreeBank. In the distributed Penn TreeBank Project CD-ROM. Linguistic Data Consortium. University of Pennsylvania
8. Sekine, S., Grishman, R.: A Corpus-based Probabilistic Grammar with Only Two Non-terminals. Fourth International Workshop on Parsing Technology. (1995)
9. The Proteus Project Parser URL: `http://nlp.cs.nyu.edu/app/`
10. J Cameron, P.: Sets, Logic and Categories. Springer. (1999)
11. Rotman, J.: Galois theory(2nd ed.). Springer. (1998)
12. Hodges, W.: A shorter model theory. Cambridge University Press. (1997)