日本語契約書の一般的な構造定義に向けて

舟木 類佳 [1,2]　末永 幸平 [2]　森 信介 [2]
[1] 株式会社 LegalForce　[2] 京都大学
ruka.funaki@legalforce.co.jp
ksuenaga@fos.kuis.kyoto-u.ac.jp
forest@i.kyoto-u.ac.jp

1　はじめに

契約書は複数の当事者が契約を交わす際に契約の内容を示すため用いられる．契約書は法的な拘束力を持ち，記述に不備があるとトラブルや訴訟問題に発展することもある．そのため，法務関係者は契約時にそれらを慎重に作成しなければならず，多大な時間がかかっているという課題がある．

契約書作成時には，必要事項を満たしているか，不利な点が無いかなどに注意を払いながら作成を行う．そうした中で，効率的な契約書作成の補助ツールとして契約書の作成・レビュー支援システムが望まれている．

契約書は条や項などの構造を持っており，法務関係者が契約書を確認する際には，契約書をそのような構造ごとで捉えることが多い．そして，レビュー支援ツールにおいては，条などの意味のまとまりごとに表示したり，構造上の単位ごとに言語処理を施し，修正案を提案したりする必要がある．

通常，契約書は Microsoft Office の Word ファイルとして作成されることが多い．Word ファイルからはテキスト情報を抽出することが可能であるが，通常このテキスト情報には契約書の構造は明示されていない．よって，効率的な契約書作成の支援のために言語処理を施すためには事前にテキスト情報を構造化することが必要になってくる．

契約書の書式は特に定まったものが存在しておらず，「契約自由の原則」により，口頭での契約を含めて自由である [4]．その一方で，契約書は慣例的に複数の代表的な書式が用いられることが多いため，大多数のものを構造化することが有益であると考えられる．

そこで，本研究では日本語の契約書とは一般的にどのようなものかを形式的に表現し，構造化することを考える．すなわち，契約書における代表的な書式パターンを定め，解析表現文法 (Parsing Expression Grammar; PEG)[1] により定義する．

2　関連研究

日本語の契約書の構造を対象にした研究はいくつか存在する．

予め契約書を XML などの構造として記述しておき，それらをもとに契約書を生成する研究は存在する [3][2]．しかし，これらの研究は契約書の書式の構造パターンを見つけることを目的にしているわけではない．

3　契約書の構造

一般的に契約書は図 1 のような形式で記述される．すなわち，契約書はタイトルで始まり，前文の後に本文が続き，後文，日付，署名（記名）押印欄となる．以下に契約書の構造の構成要素を列挙する．

タイトル： 契約書全体を簡潔に表す名詞句を記述する．契約書，覚書，合意書などという言葉を使って表現されることが多い．

前文： 契約主体を定め，どのような契約をするかの概要を記述する．多くの場合には一文からなることが多いが，複数の文から構成されることもある．前文は省略されることもある．

本文： 契約の内容が記述される．階層構造を持ち，上の階層から順に**条** → **項** → **号**というように階層構造として表現される．比較的長い契約書では条よりも上の階層として**章**という構造が用いられる．

後文： 契約書の作成枚数や，原本・写し等の作成に関して記述する．

作成日： 契約書を作成，締結した日付を記入する．

署名（記名）押印欄： 契約当事者の署名（記名），及び押印を行う．

4　日本語契約書の構造定義

本研究では契約書の構造を表すために形式文法を用いた．また，文法に解析表現文法 [1] を用いた．その

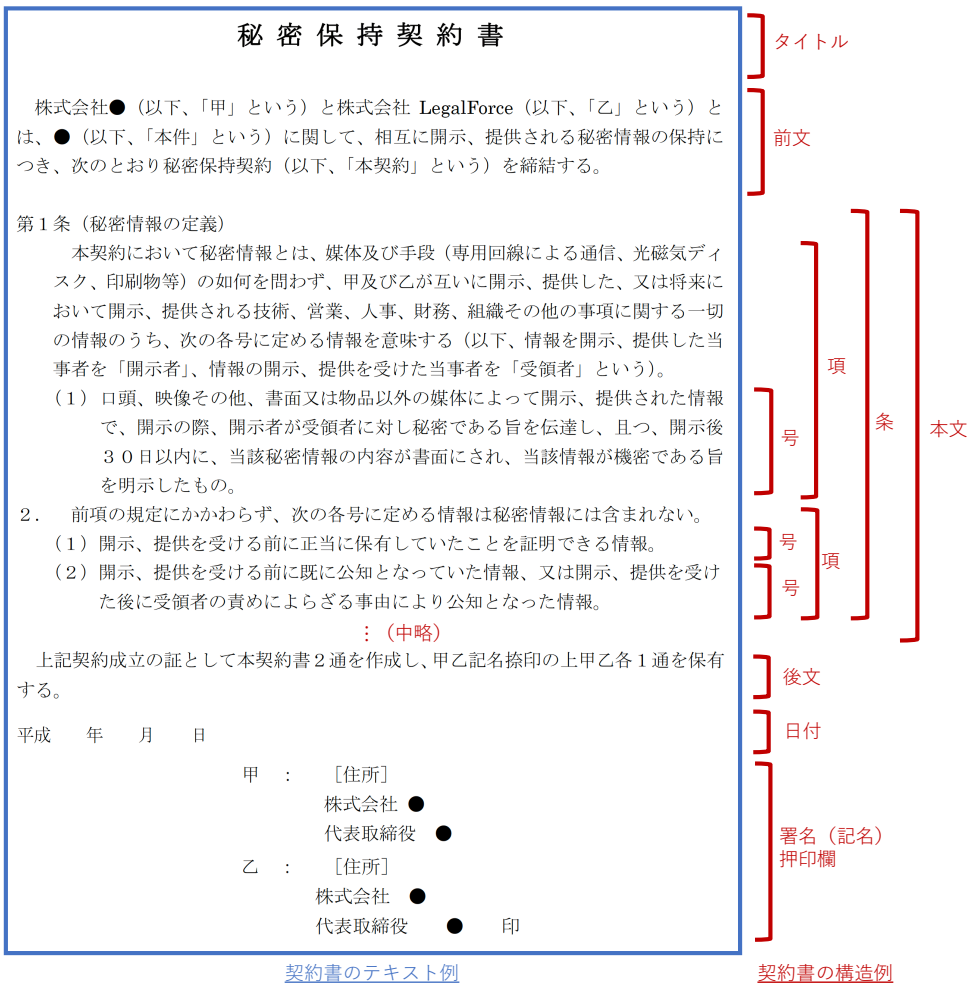

図 1: 一般的な契約書の構造の例

理由として，契約書の構造は構造上曖昧な部分は無く，曖昧性は生じ得ないと言う点が挙げられる．

4.1　解析表現文法

解析表現文法は，形式文法の一種である．文法の解析はバックトラックがなく，再帰下降構文解析に対応している．

古くから用いられているものに文脈自由文法 (Context Free Grammar; CFG) における BNF（Backus-Naur form）がある．BNF においては曖昧な文法が記述できるという問題があるが，解析表現文法では導出規則間の優先順位が一意に決まるために曖昧性の問題が生じない．

解析表現文法は $A \leftarrow e$ という形の規則の集合により定義される．ここで，A は非終端記号，e は解析表現と呼ばれる．

また，解析表現において演算子は表 1 のように表現される．

4.2　構造定義

3 節で述べたような契約書の一般的な書式に習って構造を定義した．契約書の解析表現文法の一部を図 2 に示す[1]．

図 1 の契約書を例に図 2 の構造解析を行う手順を示す．契約書の構造解析時には CONTRACT を契約書全体を表す構造とし，そこから解析表現文法に従い，順に辿っていく．まずは TITLE，すなわちタイトル（「秘密保持契約書」という文字列に対応）を探し，次に PREMISES，すなわち前文（「株式会社 . . . 」という部分に対応）を探す．そして，以降も同様に繰り返し辿っていく．

多様な書式の契約書に対応することは容易ではない．契約書の書式は揺れが生じることが多く，そのような揺れを被覆するように定義した．例えば，条文部分は図 4 のように複数のパターンが見受けられた．図の一

[1]条の中に箇条書きではない部分のことを柱書という．今回の定義では便宜上，項と柱書を同一のものとして扱っている．

```
# 契約書の定義
CONTRACT <- TITLE PREMISES ARTICLES CLOSING? SIGN_DATE SIGNATURE

# タイトルの定義
TITLE <- EMPTY_LINES? SPACES? (!END_OF_TITLE .)+ EMPTY_LINES
END_OF_TITLE <- EMPTY_LINES

# 前文
PREMISES <- EMPTY_LINES? (!END_OF_PREMISES .)* NEW_LINE
END_OF_PREMISES <- NEW_LINE  ARTICLE_TOP

# 条の定義
ARTICLES <- ARTICLE* LAST_ARTICLE?
ARTICLE <- EMPTY_LINES? ARTICLE_TOP WHITE_SPACES? ARTICLE_BODY
LAST_ARTICLE <- EMPTY_LINES? ARTICLE_TOP WHITE_SPACES? LAST_CLAUSE
ARTICLE_TOP <- ARTICLE_NUM SPACES? ARTICLE_HEADER / ARTICLE_HEADER WHITE_SPACES? ARTICLE_NUM
ARTICLE_HEADER <- OPEN_BRACKET (!CLOSE_BRACKET .)+ CLOSE_BRACKET
ARTICLE_NUM <- '第' NUMBERS_OR_PLACEHOLDERS '条'
ARTICLE_BODY <- CLAUSES

# 項の定義
CLAUSES <- FIRST_CLAUSE CLAUSE*
CLAUSE <- CLAUSE_TOP SPACES? CLAUSE_BODY
FIRST_CLAUSE <- CLAUSE_TOP? SPACES? CLAUSE_BODY
LAST_CLAUSE <- EMPTY_LINES? SPACES? (!EMPTY_LINES .)+ EMPTY_LINES
CLAUSE_TOP <- SPACES? NUMBERS_OR_PLACEHOLDERS [.. ]?
CLAUSE_TEXT <- (!END_OF_CLAUSE .)+
CLAUSE_BODY <- CLAUSE_TEXT EMPTY_LINES ITEMS?
END_OF_CLAUSE <- EMPTY_LINES (ARTICLE_TOP / CLAUSE_TOP / ITEM_TOP)

# 号の定義
ITEMS <- ITEM+
ITEM <- ITEM_TOP SPACES? ITEM_BODY
ITEM_TOP <- SPACES? OPEN_PARENTHESES NUMBERS_OR_PLACEHOLDERS CLOSE_PARENTHESES
ITEM_BODY <- (!END_OF_ITEM .)+ EMPTY_LINES
END_OF_ITEM <- EMPTY_LINES (ARTICLE_TOP / CLAUSE_TOP / ITEM_TOP)

# 後文
CLOSING <- EMPTY_LINES? SPACES? (!END_OF_CLOSING .)* EMPTY_LINES
END_OF_CLOSING <- EMPTY_LINES SIGN_DATE

# 日付
SIGN_DATE <- EMPTY_LINES? SPACES? DATE (!(SPACES? NEW_LINE) .)* SPACES? NEW_LINE
DATE <- YEAR_IN_DATE MONTH_IN_DATE DAY_IN_DATE
YEAR_IN_DATE <- (!(YEAR / NEW_LINE) .)* YEAR
MONTH_IN_DATE <- (!(MONTH / NEW_LINE) .)* MONTH
DAY_IN_DATE <- (!(DAY / NEW_LINE) .)* DAY
YEAR <- '年'
MONTH <- '月'
DAY <- '日'

# 記名（署名）押印欄
SIGNATURE <- EMPTY_LINES? SPACES? .*
```

図 2: 契約書に対する解析表現文法（PEG）の定義（一部）

番上のものは第一項に数値を省略しているが，二番目のものは数字を省略していない．また，図の一番上のものは左上から「条番号」→「条見出し」の順序であるが，三番目のものは「条見出し」→「改行」→「条番号」となっている．その他，インデントに用いる文字として，半角スペース，全角スペース，タブなどが入り乱れている部分に関しても許容した．ただし，現時点で章を持つ構造には対応していない．

5　契約書データへの適用

前節で定義した解析表現文法の定義の妥当性を検証するために実際の契約書データに対して提案する文法

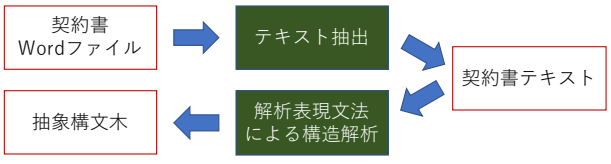

図 3: 解析表現文法（PEG）に基づく構造解析

で表現できるかを確認した．

契約書データには，Word 文書から抽出したテキストを用いた．本研究では 46 件の契約書を用いた．これらの契約書の選び方は，多様な契約書に対応できるよう，様々な契約書類型（秘密保持契約書，業務委託

表 1: PEG で利用される演算子 [1]

演算子	タイプ	優先度	説明
' '	primary	5	文字列リテラル
" "	primary	5	文字列リテラル
[]	primary	5	文字クラス
.	primary	5	任意の文字
(e)	primary	5	グルーピング
$e?$	unary suffix	4	オプション
$e*$	unary suffix	4	0 回以上
$e+$	unary suffix	4	1 回以上
$\&e$	unary suffix	3	肯定先読み
$!e$	unary suffix	3	否定先読み
$e_1\ e_2$	binary	2	シーケンス
e_1/e_2	binary	1	優先度付き選択

第１条（秘密情報の定義）
本契約において秘密情報とは・・・

第１条（秘密情報の定義）
1　本契約において秘密情報とは・・・

（秘密情報の定義）
第１条　本契約において秘密情報とは・・・

図 4: 条番号と条見出しのバリエーション

契約書，等）から選択した．

これらのデータを用いて予め定義した解析表現文法により構造解析を行った[2]．解析結果として抽象構文木（Abstruct Syntax Tree; AST）を得る（図 3）．本研究では解析表現文法の定義と構造解析に Ruby 言語のライブラリである Parslet[3]というライブラリを用いた[4]．

冒頭で述べたとおり，契約書の書き方に関しては制限がない．そのため 100%のカバー率を得ることは困難である．本研究ではより多くの契約書を構造解析できるようにカバー率を高めることを目指した．

その結果，46 件中 38 件の契約書において解析ができることを確認した（被覆率 82.6%）．解析に失敗した 8 件に関して調査したところ以下の理由に分類した．

1. 章を含む為（ 3 件）
2. 「(以下余白)」を含む為（ 1 件）
3. 契約書自体の構造が誤っている為（ 2 件）
4. 条見出しを含まない為（ 1 件）
5. 日付，記名押印欄を含まない為（ 1 件）

理由 1 は現時点で章を含む構造解析ルールを作っていないことが原因である．章を含むものは少なからず存在するので対応していきたい．理由 2，4，5 のようにイレギュラーなものも存在するが，種類によっては対応可能であると考えている．理由 3 のものについては，解析できないことは正常であると考えている．

6　おわりに

契約書のテキストを，解析表現文法により表現した．これにより，すべての書式を網羅できるわけではないものの，一般的に契約書がどのような書式や構造で書かれるかを，形式的に表現することが可能となった．

このように，契約書の構造を捉え，構造化することは，契約書レビュー支援システムを構築する上で役に立つと考えられる．例えば，条や項のような構造ごとに言語処理を施し，不適切な記述を検出したり，修正案を出力することが考えられる．

今回は日本語の契約書をターゲットに研究を行ったが，今後の発展として，英語の契約書にも広げていくことが考えられる．また，今後はより汎化性能を高めるために機械学習による構造化を行うことが考えられる．その際には本研究で示した解析表現文法の表現を元に学習用データのためのアノテーションを設計することが可能である．

参考文献

[1] Bryan Ford. Parsing expression grammars: a recognition-based syntactic foundation. *POPL 04 Proceedings of the 31st ACM SIGPLANSIGACT symposium on Principles of programming languages*, Vol. 39, No. 1, pp. 111–122, 2004.

[2] 佐野泰久, 竹下亨. XML ／ Java 技術を応用した契約書類生成管理システムの開発. 情報処理学会研究報告ソフトウェア工学（SE）, Vol. 2003, No. 63(2003-IS-084), pp. 9 – 13, 2003.

[3] 青柳悦夫, 紙屋峰貴, 中島雅博, 斉藤智夫. UNIX における契約条文等定型文書生成システム. 全国大会講演論文集, 第 48 回, データ処理, pp. 281 – 282, 1994.

[4] 田中豊. 法律文書作成の基本 : legal reasoning and legal writing. 日本評論社, 2011.

[2]契約書には「別紙」などで追加情報を含む場合があるが，今回はそれらを予め省いておいた．

[3]Parslet: `http://kschiess.github.io/parslet/`

[4]ソースコード:
`https://github.com/legalforce/contract-peg-parser`