意味役割付与テキストに対する Prolog ベースの探索木による言語パタンマッチシステム構築

小笠原崇
岡山大学大学院自然科学研究科
p28m4h8q@s.okayama-u.ac.jp

竹内孔一
岡山大学大学院自然科学研究科
takeuc-k@okayama-u.ac.jp

1 はじめに

近年，言語教育や言語研究分野の支援ツール開発と活用について多くの取り組みが為されている．それらの中でもコンコーダンスツール[1)]の開発と発展によって，言語教育や言語研究分野は大きく進展してきている [1][2] [3]．現在では表層的な言語マッチだけでなく品詞や活用形，係り受けタグなどを用いた検索機能を実現した「HASHI」や「茶器」といったコンコーダンサが提案されている [4][5][2)]．加えて，FrameNet[3)] や PropBank[4)] といった意味タグ付きコーパスの構築も進んでいると共に，機械学習の枠組みを利用した意味役割付与システムについての研究も進展が多く見られる [6] [7] [8]．

これらのタグを利用した情報抽出システムの実装例として質問応答システム Watson が挙げられる [9][10]．Watson では例えば，「Author_of(X)」といったある作品の著者を取り出すための言語パタンをあらかじめ Prolog プログラムとして定義した解答候補抽出モジュールが構築されている．この言語パタンは，文節が持つタグや形態素が持つタグを用いて著者と作品関係を表す文構造を事前に定義したもので，その文構造に一致する文からの解答抽出を実現している．このようにコンコーダンサだけではなく，質問応答システムの構築にも文中のタグを用いて言語パタンを定義する手法が提案されている [11]．しかしながら，これらのシステム構築を支援する自然言語処理ツールとしての言語パタンマッチシステムはほとんど開発されていない．

そこで本研究では，形態素タグや構文タグ，意味役割タグを利用した言語パタンに対するパタンマッチシステムの実装手法を提案する．システムでは，意味役割をタグ付けされたテキストを Prolog 述語の木構造に変換する．これにより，Prolog クエリとして定義された言語パタン構造とテキスト間の言語パタンマッチを可能にする．

2 システム設計

システムの簡易フローチャートを図 1 に示す．提案するシステムでは検索対象となるテキストにタグ付与を行い，タグに基づく検索条件とのマッチ文探索，マッチ結果出力を行う．ここでシステムの設計

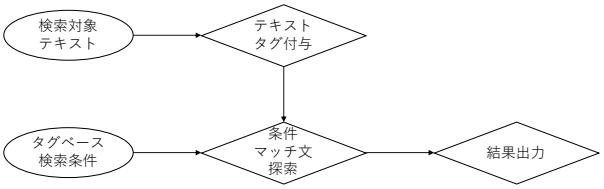

図 1 システムの簡易フローチャート

に際して，意味役割付与モジュールが必要である．実装したシステムでは，当研究室で開発を進めている意味役割付与システム ASA を用いてテキストに対してのタグ付けを行う．ASA はテキストの形態素解析，係り受け解析及び述語項構造シソーラス[5)]が分類する意味役割に基づいて意味役割付与を行うシステムである [12]．(図 2) また前述の通り，条件

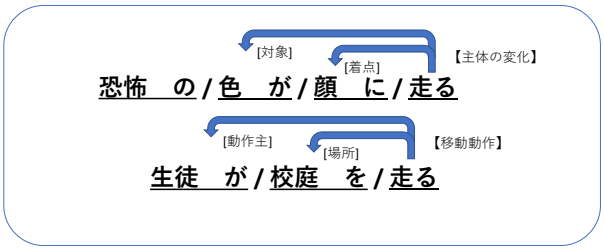

図 2 ASA の意味役割付与の例

マッチ文探索部では検索条件と検索対象テキストの

1） コンコーダンサと呼ばれることが多い．本論文でもそう呼ぶこととする．
2） NPCMJ などの構文木に対しては TGrep などの木構造ベースの探索ツールなどが提供されている http://npcmj.ninjal.ac.jp/interfaces/
3） https://framenet.icsi.berkeley.edu/fndrupal/
4） https://propbank.github.io
5） http://pth.cl.cs.okayama-u.ac.jp

Prolog を用いた解探索を行う．ここで，Prolog を用いた探索部の概要を図 3 に示す．タグ付きテキスト

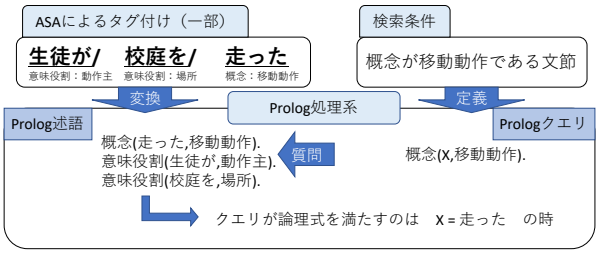

図 3　Prolog パタンマッチ部の概要

を Prolog のプログラムに変換し，Prolog のクエリ形式で定義された検索条件によって文構造の一致判定と解の抽出を行う．Prolog のプログラムはある対象物の間に存在する事実や関係性を述語として定義することで論理式として解釈できる．例えば，「花子と太郎は兄弟である」という事実は Prolog の述語で

兄弟 (花子, 太郎).

と表すことができる．この論理式に対して変数 X を用いて Prolog クエリを「**兄弟 (花子**,X).」と定義し，論理式を成立させる X が存在するかを Prolog 処理系に質問することができる．これと同様に，文節に付与される概念タグ，意味役割タグや形態素に付与される品詞タグを Prolog の述語として定義することで，検索クエリとテキスト間のマッチングを実現する．

次に，提案するシステムの全体像を図 4 に示す．提案システム全体の実装言語は Python を用いた．

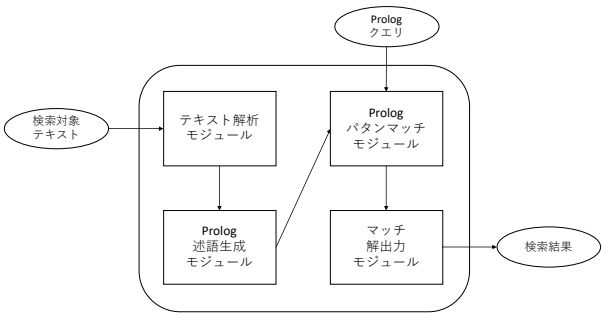

図 4　Prolog を用いた言語パタンマッチシステムの全体像

各モジュールの処理内容は以下の通りである．

テキスト解析モジュール　ASA による検索対象テキストの解析を行い，タグが付与されたテキストを生成する，

Prolog 述語生成モジュール　タグが付与されたテキストを Prolog の述語に変換し，ファイルとして出力する．

Prolog パタンマッチモジュール　Prolog ファイルと Prolog クエリを処理する．Python から呼び出す Prolog 処理系がパタンにマッチする解を探索し，解を出力する．

マッチ解出力モジュール　Prolog 処理系が出力する解を Python で抽出し，データ整形，出力を行う．

以下の節では，これら各モジュールの具体的な実装手法を説明する．

2.1　テキスト解析モジュール

テキスト解析モジュールは検索対象となるテキストを読点区切りの一文ずつ ASA により解析し，意味役割タグを付与した状態を生成する．ASA は文の文節単位，形態素単位で解析を行いタグを付与するため，解析結果はそれらをノードとする木構造で解釈することができる (図 5)．システムではこれらの

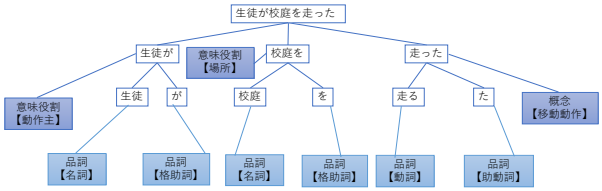

図 5　ASA による文解析の木構造

ノードの接続関係を維持し，ノード間の関係性を明確にした Prolog 述語を生成することで，タグを用いた言語パタンの定義を可能にする．

2.2　Prolog 述語生成モジュール

Prolog 述語生成モジュールではテキスト解析モジュールで生成された意味役割タグ付きテキストの木構造を基に Prolog の述語集合を生成し，ファイルに出力する．木構造のノード間の関係性を Prolog の述語名として，ノードの持つ値を述語引数として Prolog 述語を定義する．ASA の解析結果木構造から生成される Prolog 述語の一覧を表 1 に示す．ここ

表 1　Prolog の述語一覧

phrase(文, 文から分割された文節).
role(文節, 文節に付与された意味役割タグ).
semantic(文節, 文節に付与された概念タグ).
main(文節, 文節から分割された主形態素).
part(文節, 文節から分割された副形態素).
class(形態素, 形態素に付与された品詞タグ).

で，文「生徒が校庭を走った」の文節「生徒が」に対して生成される Prolog の述語集合を図 6 に示す．文「生徒が校庭を走った」は文節「生徒が」を持つ

```
phrase(生徒が校庭を走った, 生徒が).
role(生徒が, 動作主).
main(生徒が, 生徒).
part(生徒が, が).
class(生徒, 名詞).
class(が, 格助詞).
```

図 6　生成される Prolog 述語集合の例

という関係性が述語名 phrase で表現される．他も同様に，文節や形態素，付与されたタグ間の関係性が述語名で表現されることにより，Prolog 述語集合で木構造が再現される (図 7)．この Prolog 述語集合をファイルに出力し，検索クエリとともに Prolog 処理系で処理を行う．

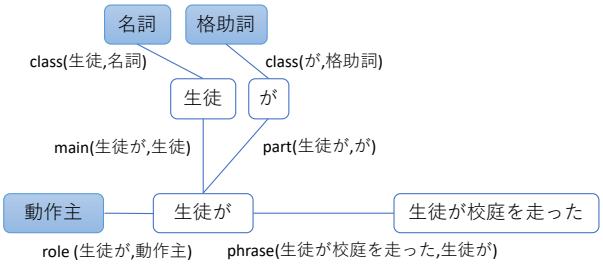

図 7　再現される Prolog の木構造

2.3　Prolog パタンマッチモジュール

Prolog パタンマッチモジュールでは，Prolog 述語生成モジュールで出力した Prolog 述語ファイルと Prolog クエリとして設定された言語パタンを用いて，Prolog によるパタンマッチを実行する．前述の通り，システム全体は Python で構築しているため，外部の Prolog 処理系を子プロセスとして起動して Prolog 述語ファイルとクエリを渡すことで，パタンマッチ結果を獲得する．本システムでは Prolog 処理系 SWI-Prolog の Python インターフェイス Pyswip モジュールを用いて構築した．

ここで，Prolog のクエリ記法について説明する．前述の通り，Prolog クエリの述語引数部分を変数で表すことにより，述語を満たすように変数が単一化[6]される．Prolog の変数は大文字のアルファベットもしくは「_」(半角アンダーバー) から始まる文字列を用いて表す．例えば，図 6 に示した Prolog 述語集合から意味役割が【動作主】である文節を抽出するクエリは，

```
role(_phrase, 動作主).
```

と表すことができる．対象となる Prolog 述語集合は，

```
role(生徒が, 動作主).
```

という述語を含むため，単一化により _phrase に**生徒が**が代入され，クエリを成立させる _phrase が述語集合内に存在しない場合は false が返される．また，変数を用いずにクエリを設定することも可能である．例えば，

```
role(生徒が, 動作主).
```

というクエリを入力すると，Prolog 述語集合内にクエリと全く同じ述語が存在するかを問い合わせることができる．存在すれば true が，存在しなければ false が返される．これらの記法と，論理和や論理積，論理否定の記法を合わせることでクエリを設定できる．クエリで用いることができる論理和，論理積，論理否定の記法を表 2 に示す．

表 2　Prolog のクエリ記法

論理和	述語 ; 述語
論理積	述語 , 述語
論理否定	not(述語)

Prolog 処理系により探索された解は処理系で出力されるが，出力データ構造が特徴的であるためマッチ解出力モジュールにてデータを整形してシステムの出力とする．

2.4　マッチ解出力モジュール

マッチ解出力モジュールでは Prolog 処理系が出力する解のデータ構造を整形してシステムの出力を行う．Pyswip モジュールは Prolog 処理系の出力を Python 順序付き辞書に変換して抽出する機能を持つため，Python から Prolog 処理系の解にアクセスすることが可能になる．この解を Python プログラムから出力した結果の例は以下の通りである．

- クエリが成立し，変数が単一化される場合の例

```
[OrderedDict([
  ('_phrase1',"生徒が"),
  ('_phrase2',"校庭を")
])]
```

- 単一化なしでクエリが成立する場合

```
[OrderedDict()]
```

- クエリが成立しない場合

```
[]
```

6)　二つの項が同一になるような変数の代入を見つけ適用すること

変数が単一化される場合，変数と代入される値の組が格納された順序付き辞書形式で出力される．変数の単一化なしでクエリが成立する場合，空の順序付き辞書が出力される．クエリが成立しない場合，空のリストが出力される．単一化が起きる場合は，ユーザが定義したクエリへの解として，Pyswip モジュールからの出力をそのままシステムの出力とする．一方で後者二つの例は，本来 Prolog 処理系が true もしくは false と出力するものであり，システムでは単にクエリがマッチする解があるか否かを出力としたい．よって，Pyswip モジュールによる出力の [OrderedDict()] を true，[] を false として変換することでシステムの出力とする．

3 システム動作確認

構築したシステムを動作させる一例を以下に示す．本確認では，検索対象とするテキストから「著者」と「作品名」のペアを抽出することとする．検索対象とする例文に以下の 8 文からなるテキストを用意した．

「有川浩が図書館戦争を書いた。学校で作文を書く。私は漫画をノートに書いた。彼は半円を書いた。彼は半円を描いた。鳥山明がドラゴンボールを描いた。母が僕を生んだ。エジソンが電球を発明した。」

これらのテキストに対し，「著者」と「作品名」を含む言語パタンを以下の Prolog クエリとして用意した．

```
author(AUTHOR,WORK):-
  semantic(_phrase1, 生成),
  role(Author_phrase, 動作主),
  role(Work_phrase, 対象),
  main(Author_phrase,_word1),
  not(class(_word1, 代名詞)),
  main(Work_phrase,_word2),
  not(class(_word2, 代名詞)),
  not(main(_phrase1, 発明する)),
  main(Author_phrase,AUTHOR),
  main(Work_phrase,WORK).
```

検索対象テキストと Prolog クエリをシステムに入力することにより，以下の結果が得られる．

```
[
  [OrderedDict([
    ('AUTHOR',"有川浩"),
    ('WORK',"図書館戦争")
  ])],
  [OrderedDict([
    ('AUTHOR',"鳥山明"),
    ('WORK',"ドラゴンボール")
  ])]
]
```

4 今後の課題

今後の課題として，このシステムを日本語にのみ対応するツールとしてではなく多言語にも対応できるよう構築を進めたいと考える．また，システムの出力に関して，言語パタンとテキストがマッチしない場合に false を出力する仕様である．しかしながら，言語パタンが何故マッチしなかったのかを考えながら言語パタンを調整したい場合，出力が false のみでは調整は容易くない．本システムが保持する Prolog 述語のテキスト木構造も出力可能にするなどの改善の余地が見られる．

5 おわりに

本研究では，意味役割付与テキストと Prolog を用いて言語パタンマッチシステムの構築を行った．今後は，多言語対応およびシステムの出力等，改善に努めたい．

謝辞

本研究の遂行にあたって JSPS 科研費 19K00552 および国立国語研究所機関拠点型基幹研究プロジェクト「統語・意味解析コーパスの開発と言語研究」の支援を受けた．

参考文献

[1] 小木曽智信, 中村壮範. 通時コーパス用 『中納言』: Web ベースの古典語コンコーダンサー.
[2] 中條清美, アントニ・ローレンス, 内山将夫, 西垣知佳子. フリーウェア WebParaNews オンライン・コンコーダンサーの英語授業における活用. 日本大学生産工学部研究報告 B (文系), Vol. 47, pp. 49–63, 2014.
[3] 藤原康弘. 小学校英語ウェブコンコーダンサーの構築と利用: 教科化を見据えて (シンポジウム 英語教育・研究のための教材コーパスの構築と利用: 実践例と課題). 英語コーパス研究, No. 22, pp. 65–76, 2015.
[4] 田中良ほか. 多言語対応コンコーダンサー『HASHI』: 日本語と日本語教育と社会言語学の研究を中心に. 2015.
[5] 松本裕治, 浅原正幸, 岩立将和, 森田敏生ほか. 形態素・係り受け解析済みコーパス管理・検索ツー

ル「茶器」. 研究報告自然言語処理 (NL), Vol. 2010, No. 18, pp. 1–6, 2010.

[6] Richard Johansson and Pierre Nugues. A FrameNet-based semantic role labeler for Swedish. In *Proceedings of the COLING/ACL 2006 main conference poster sessions*, pp. 436–443, 2006.

[7] Richard Johansson and Pierre Nugues. Dependency-based semantic role labeling of Propbank. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pp. 69–78, 2008.

[8] 竹内孔一, 土山傑, 守屋将人, 森安祐樹. 類似した動作や状況を検索するための意味役割及び動詞語義付与システムの構築. 電子情報通信学会技術研究報告. NLC, 言語理解とコミュニケーション, Vol. 109, No. 390, pp. 1–6, 2010.

[9] Adam Lally and Paul Fodor. Natural language processing with Prolog in the IBM Watson system. *The Association for Logic Programming (ALP) Newsletter*, 2011.

[10] Adam Lally, John M Prager, Michael C McCord, Branimir K Boguraev, Siddharth Patwardhan, James Fan, Paul Fodor, and Jennifer Chu-Carroll. Question analysis: How Watson reads a clue. *IBM Journal of Research and Development*, Vol. 56, No. 3.4, pp. 2–1, 2012.

[11] Kohsuke Yanai, Misa Sato, Toshihiko Yanase, Kenzo Kurotsuchi, Yuta Koreeda, and Yoshiki Niwa. Struap: A tool for bundling linguistic trees through structure-based abstract pattern. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 31–36, 2017.

[12] 竹内孔一, 石原靖弘, 竹内奈央. 述語項構造のシソーラス分類と意味役割の設計について. 人工知能学会全国大会論文集 第 27 回全国大会 (2013). 一般社団法人人工知能学会, 2013.