研究速報

依存構造に基づくコーパス検索

加藤　芳秀[†a)]（正員）　松原　茂樹[††]（正員）
稲垣　康善[†††]（名誉員：フェロー）

Corpus Search Based on Dependency Structure

Yoshihide KATO[†a)], Shigeki MATSUBARA[††], *Members*, and Yasuyoshi INAGAKI[†††], *Fellow*

† 名古屋大学大学院国際開発研究科，名古屋市
Graduate School of International Development, Nagoya University, Nagoya-shi, 464–8601 Japan
†† 名古屋大学情報連携基盤センター，名古屋市
Information Technology Center, Nagoya University, Nagoya-shi, 464–8601 Japan
††† 愛知県立大学情報科学部，愛知県
Faculty of Information Science and Technology, Aichi Prefectural University, Aichi-ken, 480–1198 Japan
a) E-mail: yosihide@gsid.nagoya-u.ac.jp

あらまし　キーワード系列の入力に対して，系列中のキーワードがコーパス内の各文において形成する依存構造パターンを同定し，パターンごとに検索結果を分類する用例文検索を提案する．文の構文的構造を活用した用例文検索が実現できる．

キーワード　用例文検索，依存構造，構文木付きコーパス，依存構造パターン

1. まえがき

近年，言語資源としての大規模コーパスの重要性はますます高まっており，言語現象の調査，外国語学習，自然言語処理システムの開発など様々な場面で活用されている．コーパスを効果的に活用するために，様々なコーパス検索システムが提案されている．

従来システムの多くは，いくつかのキーワードをクエリとして受け取り，それらを含む用例文を検出する．精度の高い検索を実現するために，クエリ中の隣接するキーワード間の距離を定義し，ある距離以下でキーワードが出現する文のみを検出する手法なども提案されている [5]．キーワードに基づく検索は，直観的でユーザビリティが高い反面，構文構造などの言語的構造を活用した検索はできず，ユーザは検出された文においてキーワード同士にどのような関係があるのかを知ることはできない．

一方で，構造的な条件をユーザが明示的に指定して検索するシステムがこれまでに提案されている [2], [3], [6]．これらのシステムでは，句構造や依存構造のパターンをクエリとしてコーパスを検索する．

Corley らの提案する Gsearch [2] では，入力として，句構造パターンを生成するための文法と検出したい句構造パターンを受け取る．システムは入力された文法を用いてコーパス中の文を構文解析し，与えられた句構造パターンをもつ文を検出する．

兵藤らの手法 [3] や，Resnik らの提案する Linguist's Search Engine（LSE）[6] では，ユーザはまず，検出したい文の一例を入力する．入力された例文は構文解析され，結果が提示される．ユーザは提示された構文解析結果を編集し，構造的なパターンを作成する．最終的にシステムは，編集により得られたパターンにマッチする構造をもつ文をコーパスから検出する．

しかしながら，これらのシステムが，キーワードに基づく検索のような，簡単で直観的なインタフェースを実現しているとは言いがたい．文法の設計や句構造パターンの編集が可能なレベルの文法的知識をもたないユーザは，これらのシステムを利用することはできない．

そこで本論文では，キーワード系列をクエリとして受け取り，それらが形成する構造的なパターンを自動的に同定する用例文検索を提案する．本手法では，言語構造として依存構造を用いる．検索対象のコーパスには，依存構造解析器などによりあらかじめ依存構造が付与されていることを前提とする．クエリを受け取ると，まずコーパス内を検索し，クエリ中のすべてのキーワードをその順序どおりに含む文を検出する．次に，検出された各文において，キーワード系列が形成する依存構造パターンを，コーパスに付与された依存構造を参照して同定する．

文の依存構造に応じて，キーワード系列は異なった依存構造パターンを形成することになる．本手法では，依存構造パターンごとに文を分類し，依存構造パターンとそれにマッチする依存構造をもつ文を併せて表示する．依存構造パターンを提示することにより，ユーザは表示された文においてキーワード同士がどのような構文的関係にあるかを知ることができる．また，特定のパターンをもった文を探す場合にも，ユーザはいくつかの依存構造パターンの中から必要とするパターンを選択するだけでよく，文法の設計や構造的なパターンの編集などは不要である．

2. 依存構造パターン同定アルゴリズム

本章では，提案するコーパス検索の中心をなす依存構造パターン同定アルゴリズムについて述べる．同定された依存構造パターンに従って，コーパス中の文は分類される．本アルゴリズムは，キーワード（単語，

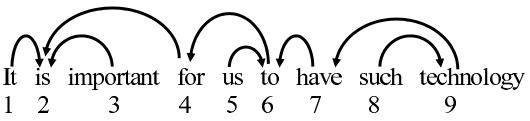

図 1 “It is important for us to have such technology” に対する依存関係
Fig. 1 Dependency relations for a sentence “It is important for us to have such technology.”

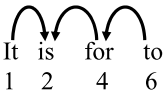

図 2 依存構造パターンの例
Fig. 2 An example of dependency structure pattern.

または品詞(注1)）の系列，文，及び文の依存構造が与えられると，キーワード系列が形成する依存構造パターンを同定する．依存構造パターンは，キーワードに対応する文中の単語の間に成り立つ依存関係を表現する．例えば，キーワード系列

$$\text{it is for to} \tag{1}$$

及び，図 1 の文と依存構造が与えられると，図 2 の依存構造パターンを出力する(注2)．この依存構造パターンは，各キーワードに対応する単語 “It,” “is,” “for,” “to” の間の依存関係（例えば，“It” が “is” に依存するという関係）を表現している．

本アルゴリズムは，キーワード系列からボトムアップに依存構造パターンを生成することにより，キーワード系列全体が形成する依存構造パターンを同定する．生成される依存構造パターンは，文の依存構造にマッチするパターンのみである．

入力として，

クエリ $q_1 \cdots q_m$（q_i $(1 \le i \le m)$ はキーワード）

文 $s = w_1 \cdots w_n$ （w_j $(1 \le j \le n)$ は単語と品詞の対）

依存構造（= 依存関係の集合） D

を受け取り，依存構造パターンの集合を出力する．ここで，D は，文 s に出現する単語間の依存関係の集合である．w_j が w_k に依存するとき，その単語位置の対 (j,k) は D の要素である．以下では，(j,k) を $j \to k$ と書くことにする．

依存構造パターンは 3 項組 $d = (h, L, R)$ である．h は単語位置であり，これを d の主辞と呼ぶ．L，及び R は，依存構造パターンのリストである．(h, L, R) は，L 中の依存構造パターンの主辞が左から h に依存することを意味する．R の場合は，右から依存することを意味する．

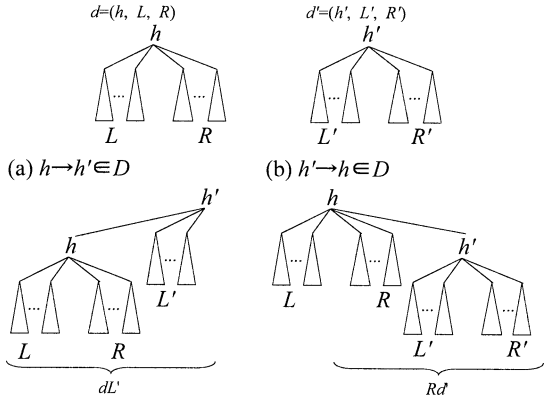

図 3 結 合 操 作
Fig. 3 Combining.

本アルゴリズムは，クエリ $q_1 \cdots q_m$ に対して以下の操作をボトムアップに適用することにより，クエリ中のキーワードから構成される依存構造パターンを生成する．クエリ全体に対して依存構造パターン d が生成されるとき，s の依存構造が d とマッチすることを意味する．

初期化 各 q_i $(1 \le i \le m)$，w_j $(1 \le j \le n)$ に対して，q_i が単語 w_j あるいはその品詞にマッチするならば，q_i に対する依存構造パターンとして $(j, \varepsilon, \varepsilon)$ を生成する．

結合操作 $d = (h, L, R)$，及び $d' = (h', L', R')$ をそれぞれ，$q_i \cdots q_j$，及び $q_{j+1} \cdots q_k$ に対する依存構造パターンとし，d 中の最も右に出現する単語位置が，d' 中の最も左に出現する単語位置よりも小さいものとする．$h \to h' \in D$ かつ $R' = \varepsilon$ ならば，$q_i \cdots q_j q_{j+1} \cdots q_k$ に対して，依存構造パターン (h', dL', R') を生成する（図 3 (a) 参照）．$h' \to h \in D$ ならば，依存構造パターン (h, L, Rd') を生成する（図 3 (b) 参照）．

結合操作を繰り返し適用することにより，クエリ中の各キーワードが，クエリ中の別のキーワードに直接依存するようなすべてのパターンを生成できる．すなわち，パターンが生成された文のみが，クエリ中のキーワードが依存関係にある文である．

図 4 に依存構造パターン同定アルゴリズムを示す．

(注1)：単語や品詞の文字列パターンを正規表現により記述することもできる．
(注2)：英語文を例に考えるが，本手法は，依存構造が付与された文であれば言語によらず適用可能である．

```
input: query q_1 ··· q_m,
       sentence w_1 ··· w_n,
       dependency structure D

initialization:
for i = 1 to m
  for each j s.t. w_j = q_i do
    push (j, ε, ε) to D[i − 1, i];

combining:
for k = 2 to m
  for j = k − 1 down to 1
    for i = j − 1 down to 0
      for each d = (h, L, R) ∈ D[i, j],
               d' = (h', L', R') ∈ D[j, k]
               s.t. rm(d) < lm(d') do
        if h → h' ∈ D ∧ R' = ε then
          push (h', dL', R') to D[i, k];
        if h' → h ∈ D then
          push (h, L, Rd') to D[i, k];

return D[0, m];
```

図 4　依存構造パターン同定アルゴリズム
Fig. 4　An algorithm of identifying dependency structure patterns.

$D[i,j]$ は，過程で生成される $q_{i+1}\cdots q_j$ に対する依存構造パターンを記録する．

$rm(d)$ は，d 中の最も右に出現する単語位置，$lm(d')$ は，d' 中の最も左に出現する単語位置を表す．

2.1　依存構造パターン生成例

本章の冒頭に挙げた例，すなわち，クエリ (1)，及び図 1 の文とその依存構造に対する依存構造パターンの生成を考える．

まず，クエリの 1 番目の単語 “it” は，文の 1 番目の単語にマッチするので，これに対して依存構造パターン

$$(1,\varepsilon,\varepsilon) \tag{2}$$

を生成する．同様にして，“is,” “for,” 及び “to” に対してそれぞれ，

$$(2,\varepsilon,\varepsilon) \tag{3}$$

$$(4,\varepsilon,\varepsilon) \tag{4}$$

$$(6,\varepsilon,\varepsilon) \tag{5}$$

を生成する．

隣接する依存構造パターン (2) と (3) の主辞に注目すると，依存関係 $1 \to 2$ が成り立つ．したがって，“it is” に対して，依存構造パターン

$$(2,(1,\varepsilon,\varepsilon),\varepsilon) \tag{6}$$

を生成する．同様にして，“for to” に対して，

$$(4,\varepsilon,(6,\varepsilon,\varepsilon)) \tag{7}$$

を生成する．更に，$4 \to 2$ であるので，依存構造パターン (6) と (7) を結合し，“it is for to” に対する依存構造パターンとして，

$$(2,(1,\varepsilon,\varepsilon),(4,\varepsilon,(6,\varepsilon,\varepsilon))) \tag{8}$$

を生成する．以上より，図 1 の文には，依存構造パターン (8) の形で，“it is for to” が出現していることが分かる．一方，クエリ中のキーワードが依存関係が成り立つ形で出現していない文（例えば，“It is clear whether support for the proposal will be broad enough to a serious challenge”）に対しては，結合操作によって依存構造パターンは生成されない．

3.　実装と評価

提案手法を，CMUCL (注3) を用いて実装した．システムでは，検出された文は，マッチする依存構造パターンごとに分類され，依存構造パターンはグラフィカルに表示される．検索画面を，図 5 に示す．画面上部は，キーワード系列の入力部である．検索結果は，依存構造パターンの下に，それにマッチする依存構造をもつ文を並べる形で表示される．

以下では，いくつかのクエリに対するシステムの具体的な動作について考察する．検索対象として，英語新聞記事に句構造を付与したコーパスである Penn Treebank [4] の 49208 文を使用した．コーパス中の句構造は，文献 [1] の方法に従って依存構造に変換した．

クエリ中のキーワードをその順序どおりに含む文を，人手により正解・不正解に分類し，検索システムの分類とどのような関係にあるかを調べた．

動作例 1：“it is for to” に対する検索

クエリ “it is for to” に対する検索結果を表 1 に示す．クエリ中のキーワードをこの順序どおりに含む文は，合計 77 文あり，そのうち 17 文について依存構造パターンが同定された．同定された依存構造パターンはすべて，**2.1** の例と同様の形のパターンであり，そのパターンをもつ文はすべて正解であった．依存構造パターンが同定されなかった正解データを調べたところ，それらに対する依存構造では，“for” は be 動詞ではなく，補語に依存していた．このように，キーワードが他のキーワードに直接依存しないような文に対し

(注3)：http://www.cons.org/cmucl/

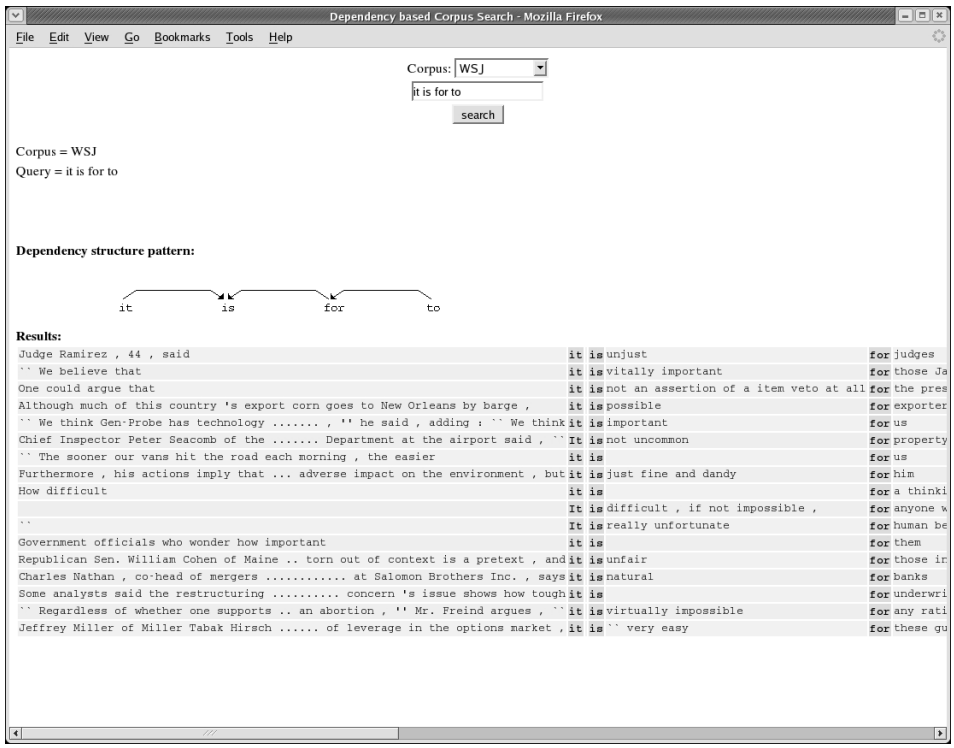

図 5 検索画面
Fig. 5 A view of sentence retrieval.

表 1 クエリ “it is for to” に対する検索結果
Table 1 Result for a query “it is for to.”

	正解	不正解	合計
パターンが同定された文	17	0	17
されなかった文	4	56	60
合計	21	56	77

表 2 クエリ “have 前置詞 mind” に対する検索結果
Table 2 Result for a query “have *preposition* mind.”

	正解	不正解	合計
パターンが同定された文	10	0	10
されなかった文	1	11	12
合計	11	11	22

てはその依存構造パターンを同定することはできないが，全体としては，81.0% (= 17/21) の正解データを誤りを含むことなく分類することに成功している．

動作例 2：“have 前置詞 mind” に対する検索

クエリ “have 前置詞 mind” に対する検索について考える．結果を表 2 に示す．このクエリについても，依存構造パターンが同定されたか否かで文を分類することにより，正解と不正解を高い精度で分類できる．

前置詞にマッチした 10 個の単語のうち，9 個は “in” であり，1 個は，“on” であった．キーワードに品詞を利用することにより，それが具体的にはどのような単語として現れるかを調べることができる．

4. むすび

本論文では，キーワード系列から依存構造パターンを同定し，同定されたパターンに基づき用例文を分類するコーパス検索システムを提案した．提案システムの有用性を，いくつかのクエリに対する具体的な動作の観察を通して確認した．結合操作のみによる依存構造パターン同定では，クエリ中のキーワードが，他のキーワードと直接には依存関係をもたない文を検出できない．このような文を検出する方法については稿を改めて論じたい．

今後の課題として，テストセットの量を増やした評価実験や，ユーザビリティの観点からの評価が挙げられる．

謝辞 本研究の一部は，名古屋大学学術振興基金，並びに科学研究費基盤研究（A）(2)（課題番号：1620001），若手研究（B）(課題番号：17700145）の助

成を受けています.

文　献

[1] M. Collins, Head-driven statistical models for natural language parsing, PhD Dissertation, University of Pennsylvania, 1999.

[2] S. Corley, M. Corley, F. Keller, M. Crocker, and S. Trewin, “Finding syntactic structure in unparsed corpora: The Gsearch corpus query system,” Computers and the Humanities, vol.35, no.2, pp.81–94, 2001.

[3] 兵藤安昭，河田実成，応江　黔，池田尚志，“構文付きコーパスの作成と類似用例検索システムへの応用,” 自然言語処理，vol.3, no.2, pp.73–88, 1996.

[4] M.P. Marcus, B. Santorini, and M.A. Marcinkiewicz, “Building a large annotated corpus of English: The Penn Treebank,” Computational Linguistics, vol.19, no.2, pp.310–330, 1993.

[5] 難波英嗣，森下智史，相沢輝昭，“論文データベースからのイディオム用例検索,” 情処学研報，NL-170, pp.53–59, 2005.

[6] P. Resnik and A. Elkiss, “The linguist’s search engine: An overview,” Proc. ACL Interactive Poseter and Demonstration Sessions, pp.33–36, 2005.

（平成 18 年 5 月 13 日受付，7 月 31 日再受付）