構文木を含むタグ付きコーパス参照支援ツール

鈴木　しとな*　上原 徹三**　石川 知雄**
*武蔵工業大学大学院工学研究科　**武蔵工業大学工学部

1　はじめに

1.1　背景

品詞や構文情報を含むタグ付きコーパス[1][2][4]は形態素解析や構文解析などにおいて、文法規則の学習や検証のため、また、解析結果の評価のためなどの目的で広く利用されている。日本語では、日本電子化辞書作成のEDR日本語コーパスが広く用いられている[1]。これは約２２万文の文例に対し、文を構成する単語に関する読み、表記、品詞、活用形などの形態素情報と、単語間の合成関係を単語を葉に持つ構文木をリストで表している構文情報などを保持している。

しかし、EDR日本語コーパスの複雑な階層構造を含む表現を解読することは容易でない。文法規則の学習と検証および解析結果の評価のためにこのEDR日本語コーパスを全体的に調査する際には、調査者は、そのデータ形式を把握し、調査プログラムの中にそれを解読する機能を実現する必要がある。しかし、この処理は煩雑である上に、調査者の本来の目的にとっては余計な処理である。

そこで、このEDR日本語コーパスを解読する機能を、調査者の必要に応じて呼び出せる関数群から成るツールを用意することとした。これが、本研究の主題であるEDR日本語コーパス参照支援ツールである。コーパス調査者は自作の応用プログラムに本来の調査目的の機能を実装する。この中で、コーパスの情報の取り出しについては本ツールの関数を用いることができる。また、手続き型の応用プログラムを作成することなく単純なコーパス調査を行いたい調査者のために、本ツールの上に文構造照合支援システムを作成することもできる[3]。

参照支援ツールの機能と適用性については一定の評価ができた[5]が、現在は、常にコーパスのレコードをレコード番号順に調べる処理を前提としているために低速である。文構造照合支援システム[3]のようなアプリケーションでは、一般的な利用のされ方を考えれば、即時対話性を持つことが望まれるが、それは本参照ツールが低速であるために実現できていない。そこで、本研究では、一般に利用される率が高い品詞による参照、表記による参照について高速化によってコーパス調査の効率を向上することを目的とする。

2　EDR日本語コーパスについて

EDR日本語コーパス[1]は20万文余りの文例を収めた大規模コーパスで、文例自身の他に品詞、意味概念などの形態素単位の情報や、係り受け関係を示す構文情報などのタグ情報を保持しているという特徴を持つ。このコーパスに含まれる一つの文例に対する保持内容を図1に示す。図のような文例一つに対する各情報のまとまりを、以下ではレコードと呼ぶこととする。

１レコード分の情報は大きく分けて、文情報、構成要素情報、形態素情報、構文情報、意味情報、および、管理情報によって構成されている。各レコードの先頭にはレコードの順番を示すレコード番号(図1の"JCO0000666"の部分)が一意につけられている。

3　EDR日本語コーパス参照ツール

EDR形式日本語コーパス参照支援ツールは、C++言語で記述された応用プログラム内から呼び出される関数群で、品詞や表記、文などのEDR日本語コーパスに記載されている情報の参照をレコード単位で行い、レコード、形態素などの単位で調査位置を移動することができる。また、構文情報で提供される単語間係り受け関係を利用し、ツール内で文節間係り受けを作成している。作成した文節については、係り先、係り元それぞれの品詞、表記などの参照、文節単位での調査位置の移動が行える。

以下に主な関数のうち本研究に関連する機能について説明する。以下の関数の他に、参照支援ツールによって作成された文節に関する情報を参照する文節用関数、コーパスレコード内の

```
JC00000666      00050000f4aa    朝日新聞８７０２０８    １０万円はそのままにしている。
{       1       １０万  １００００      数字    00010186a0      2       円      エン
接尾語  3bcca4  3       は      ハ      助詞    2621d5  4       そのまま        ソノママ
        名詞    0fb1d8  5       に      ニ      助詞    2621d5  6       し      シ
動詞    "＝Ｚ  維持する"        7       て      テ      助詞    2621d5  8       い
イ      動詞    ""      9       る      ル      語尾    2621d0  10      。      。
記号    2621d8  }       /1:１０万/2:円/3:は/4:そのまま/5:に/6:し/7:て/8:い/9:る/10:。/
(S(t(M(S(t(S(W 1 "１０万")(t(W 2 "円"))))(W 3 "は"))(t(M(S(t(W 4 "そのまま"))(W 5 "に"))
(t(S(t(W 6 "し"))(W 7 "て")(W 8 "い")(W 9 "る")))))))(W 10 "。"))      [[main 6:し:"＝
Ｚ  維持する"][attribute state][object [[main 2:円:3bcca4][number 1:１０万:00010186a0]]]
[manner 4:そのまま:0fb1d8]]     DATE="95/7/12"
```

図 1: EDR 日本語コーパスのレコードの例

調査点を移動するポインタ操作用関数等が存在する。関数は従来は41個であった[5]が、高速化のために3個増やして44個となった。

3.1 ファイル設定用関数

ファイル設定用関数には、コーパスファイルの読み込み、レコード番号をキーとするインデックスファイルの読み込み、レコード番号による参照開始位置のセットなどを行なう関数がある。

3.2 形態素要素用関数

形態素要素用関数には、カレント形態素の構成要素番号、かな表記、品詞、概念、係り先の構成要素番号、かな表記、品詞、概念のそれぞれを戻り値に持つ関数がある。

4 EDR 日本語コーパス参照ツールの高速化

現在、参照支援ツールで参照を行う場合、

”「やや」という副詞を出力する”

というようなもっとも単純な内容の参照でも、調査範囲を全体とすると、平均で7分強の実行時間を必要とする。これは参照支援ツールがコーパスの全レコードをレコード番号順に探索しているためであり、この時間は、参照項目に関わらずほぼ一定である。しかし、20万余りレコード中に、副詞の含まれるレコードは58426レコードであり、例えば、表記「やや」の表記を含むレコードは232レコードしかない。よって、あらかじめ調査するレコードを限定することにより、実行時間を大幅に削減できると考えられる。本研究ではこれらの品詞や表記によるインデックスを作成し、これを利用した高速化を行うこととする。

4.1 インデックスの作成

もともと、EDR 日本語コーパス参照支援ツールでは、レコード番号をキーとしたインデックス機能を提供している。このインデックスは、B+ツリーとして構成し、その葉ノードのインデックスレコードが個々のコーパスレコードを指す。このインデックスファイルはあらかじめ作成しておき、ツール内の関数を用いて検索できる。

そこで、高速化のために、このレコード番号を一次キーとして利用し、下記の検討によって定める品詞や表記のような参照項目を2次キーとしたインデックスを作成することとした。

4.1.1 インデックスの項目の決定

インデックス項目を決定するために、学会誌や研究会などで発表された EDR 日本語コーパスを利用している研究を対象に調査を行った。

その結果、参照条件として品詞と表記が多く使用されることがわかった。そこで、インデックスの項目として、品詞と表記を採用し、インデックスの作成、参照機能の実装を行うことにする。

4.1.2 品詞のインデックス

品詞に関しては次のようなインデックスを作成する。

インデックスは配列構造で該当するレコード番号を保持することとした。保持するレコード番号に要するメモリ量の削減のため、高頻度の品詞については、それが出現しないレコードのレコード番号を持つこととした。そこで、そのレコード番号がその品詞が出現するレコードを示すのか、品詞が出現しないレコードを示すのか

を表すフラグ(このフラグを有/無フラグと呼ぶ)を各配列に持つこととした。EDR 日本語コーパスの文例の内、参照ツールが対象としている文における主要な品詞の出現文数と対象の全レコードに占める割合を表1に示す。「レコード数」は各品詞を含むレコード数で、そのレコードが全レコードに占める割合を「含む率」として示している(全文は 203687 文である。これは EDR 日本語コーパスの全レコード数より少ないが、文節間係り受けに交差がある文を除いたためである)。

表 1: 主な品詞がコーパスに含まれる率

品詞	レコード数	含む率(%)
名詞	203,461	99.9
動詞	198,634	97.5
形容詞	45,301	22.2
副詞	58,426	28.7
助詞	203,594	99.9

名詞、動詞、語尾、助詞、助動詞、記号については、含まれる文数が多いため、出現しないレコードをデータとして持ち(有/無フラグ='無')、残りの形容詞、形容動詞、副詞、連体詞、接続詞、接頭語、接尾語、感動詞、数字については、出現するレコードをデータとして持つ(有/無フラグ='有')こととした。

4.1.3 単語表記のインデックス

単語の表記の場合、キーとなりうる語が非常に多く、データ量が莫大になるため、キーとなる表記をあらかじめ限定する。まず、記号、数字、名詞を品詞として持つ表記は、参照条件となる可能性が低いためインデックスから除く。次に、高速化される割合とデータの量のバランスを考え、一定の閾値を設け、表記を限定することでデータの量を抑える。単語の表記は活用語の場合、語幹と語尾に分けて持つ。1次キーは品詞の場合と同様にレコード番号とする。

表記のインデックスにおける閾値を決定するために、各表記の出現頻度の調査を行った。その結果、表記が含まれるレコード数の分布は非常に偏っていることが分った。コーパス全文(203,687文)に含まれる単語の異なり表記は 120,402 表記で、そのうち、120,096 表記が出現レコード数 1000 回以下に集中している(1レコード内に同一の表記がいくつ出現しても1回とカウント)。全レコードの約半数にあたる 100,000 レコードを超える表記はごく少数である(表2)。表中の表記のうち"る"および"い"は主に活用語尾の働きをする表記で、それ以外は主に助詞、助動詞となる表記である。

表 2: 出現頻度の高い表記

表記	含まれるレコード数
に	127,061
る	108,704
が	100,124
い	109,425
は	131,043
を	110,627
の	152,911

出現頻度の調査結果を検討し、該当する表記の品詞、利用頻度などを考慮した結果、20,000 レコードを閾値とし、これより出現頻度の低い表記をキーとすることにした。

4.2 インデックスによる参照用関数

品詞および単語の表記による参照条件を指定するために、関数側に要求される機能について検討する。関数は、品詞インデックス用、単語の表記インデックス用をそれぞれ作成する。参照項目の入力では、参照したい項目を引数として与え、複数の項目を入力する場合には'|' で区切る。また、入力の終了を示す終了記号として'$'を記入する。また、関数を組み合わせて使用する場合には、入力が継続する場合には'&'を記入する。

例 副詞の参照

```
SubjectHinshi("副詞 | $");
```

4.2.1 要求される機能およびその実現方法

基本的な要求機能は、次の3種ある。

- ある品詞/表記を含むレコードの参照
- ある品詞/表記を含まないレコードの参照
- 上記の参照が複数個ある場合、それらの AND/OR による参照

AND と OR の指定については、同種の項目の場合には OR が多く、品詞と表記を組み合わせる場合には AND が多いと考えられるので以下のようにする。

- 各参照用関数とも、引数に複数の項目を与えた場合、OR 指定とする。

例 形容詞または形容動詞の参照

SubjectHinshi("形容詞 | 形容動詞 | $");

- 各参照用関数を、同じ関数または異なる関数を組み合わせて使った場合、AND 指定とする。

例 「やや」という副詞

SubjectHyoki("やや | &");
SubjectHinshi("副詞 | $");

- 各参照用関数を組み合わせて複数回使った場合、各参照用関数の組合せの OR 指定とする。

さらに複雑な論理式の実現は、以上の基本機能と組み合わせてユーザ作成の応用プログラムで実現するものとする。

4.2.2 否定の指定

文法調査においては、"「やや」でない副詞"のように、否定によって条件を指定する場合が考えられる。そこで、否定の場合の参照について機能を用意する。指定の方法としては、関数に品詞および単語の表記を引数として与えるときに、否定の場合、'!' を頭につけることで、区別する。

例 「やや」でない 副詞

SubjectHyoki("!やや | &");
SubjectHinshi("副詞 | $");

5 インデックスファイルの作成

インデックスファイルの作成には、CPU intel Celeron 433MHz、メモリ 256MB のマシンを使用した。OS は FreeBSD Version4.0、プログラムは C++で記述した。品詞のインデックスは各品詞 (作成した品詞は全部で 15 種) について約 7 分で作成、配列で構成したインデックスファイルの量は 2,652,246 バイトであった。単語表記 (キーとした表記は 17,811 個) のインデックスでは、作成に約 3 時間 30 分かかり、B+ツリーの葉ノードから指すインデックス情報の量は、6,558,765 バイトであった。

6 結言

本研究では、構文木を含むタグ付きコーパスの参照支援ツールに関して高速化を行った。そのために一次キーをレコード番号とし、2 次キーとして次の 2 種類を対象とするインデックスを作成した。

- 品詞をキーとする時は、出現するレコード数の少ないものに関しては出現するレコードを持ち、出現するレコードが多いものは出現しないレコード番号を配列構造として持たせた。
- 表記をキーとするときは、出現頻度に関する閾値によるキーの選別を行い、B+ツリーとして実現した。

また、参照関数では、機能的な要求を検討し、AND、OR、否定の機能を作成した。

今後はインデックス情報の圧縮や参照関数の効率化実現 [6] などの検討が必要である。また、実際の目的に適用しつつ、参照条件の指定法の適用性の評価も行う予定である。

なお、本研究の一部は文部省科学研究費補助金 (基盤研究 C 2 No.11680422) によって実施したものである。

参考文献

[1] 日本電子化辞書研究所. (1995).EDR 電子化辞書仕様説明書, 日本電子化辞書研究所.

[2] 黒橋禎夫, 長尾真 (1997). 京都大学テキストコーパスプロジェクト. 自然言語処理学会第 3 回年次大会.pp.115-118

[3] 渡部憲二, 上原徹三, 石川知雄 (2001). EDR 日本語コーパスを対象とする文構造照合支援システムの試作. 電子情報通信学会論文誌. VOL.J84-D-II No.1 pp.139-149.

[4] The Pen Treebank Project. http://www.cis. upenn. edu/treebank/.

[5] 鈴木 しとな, 渡部 憲二, 上原 徹三, 石川 知雄 (1999).EDR 日本語コーパス参照支援ツール. 情報処理学会研究報告.99-CH-44.

[6] I. H. Witte, A. M .Timothy, C.Bell(1999). Managing Gigabytes. MORGAN KAUFMANN PUBLISHERS.