大規模日本語文法の開発に関する諸問題

野呂 智哉　八木 豊　橋本 泰一　徳永 健伸　田中穂積

東京工業大学 大学院情報理工学研究科
{noro,yutaka,taiichi,take,tanaka}@cl.cs.titech.ac.jp

1 はじめに

入力文の構文構造を分析するためには文法が必要となる。この文法は，小規模なものであれば人手でトップダウンに開発できるが，分析可能な表現が限られ，網羅的でなく実用的でない。また，多様な言語現象を扱うためには大規模な文法が必要であるが，これをトップダウンに開発することは困難である。一方，大規模な構文構造付きコーパス（以下，単に「コーパス」と略す）があれば，そこから大規模な文法を抽出することでボトムアップに開発できる。Charniak[2]は，Penn Treebank コーパスから抽出した英語文法を使用し，人手で作成した文法よりも，特に単語数の多い文において精度のよい解析結果が得られることを明らかにしている。一方，日本語では Penn Treebank コーパスのような大規模なコーパスが少ない。白井ら [9] は括弧付きコーパス（EDR コーパス）に対して，括弧に与える構文ラベルを自動的に推測し，そこから文法を自動獲得する手法を提案している。しかし，自動的に推測した構文ラベルが言語学的に正しいとは必ずしも言えない。適切な大規模文法を開発するためには人手で構築した構文構造付きの大規模なコーパスを出発点とする必要がある。

しかし，大規模なコーパスから抽出した文法を使用して構文解析を行うと，非常に膨大な量の解析木を生み出すという問題がある。その原因として，人手で構築したコーパスに誤りが含まれ，そこから抽出した規則には構文解析結果の曖昧性を増大させるものが含まれることが挙げられる。これに対し，Charniak は，コーパス中における出現頻度の低い規則を削除するなどして解析精度の向上を図っている。ところが，曖昧性を無意味に増大させる文法規則の出現頻度が必ずしも低いとは限らず，出現頻度の低い規則を削除するだけでは十分ではない。

また，コーパス作成者は意味を考慮して構文構造を付けるが，そのコーパスから抽出した文法もやはり意味を考慮したものになる。この文法を使用すると，意味的な情報を使用しない構文解析では解決できない構造の曖昧性を解析結果として出力する。例えば，英語の PP attachment 問題において，前置詞句の係り先の解決は意味的な情報を使用しない限り難しい。この解決の難しい曖昧性を構文解析結果に含めることは，膨大な数の構文解析結果を生み出すことが普通であり，解析時間や使用メモリ量の増大を招くだけでなく，その後の意味解析の観点からも望ましいことではない。このような場合には，意味的な情報を使用しない限り解決の難しい構造の曖昧性を極力抑えられるようにコーパスの構造を変更し，改めて抽出した文法で構文解析を行うべきである。そして，意味的な情報を用いなければ解決できない厳密な構造解析はその後の意味解析に任せるというアプローチを採用すべきである [4, 5, 6] [1]。

以上のことから，我々は人手で構文構造を付与したコーパスを出発点とし，以下の手順でボトムアップに構文解析のための文脈自由文法（CFG）[2]を開発することを試みている [7]。

1. 既存の構文構造付きコーパスから文法を抽出
2. 構文解析木の曖昧性を増大させる文法規則の分析
3. 分析結果に基づき，構文構造付きコーパスを変更
4. 変更した構文構造付きコーパスから文法を再抽出
5. (2)〜(4) を繰り返す

これまでにコーパスから抽出して得た大規模文法は，実際にはほとんど使われていない [1]。その最大の原因は解析結果の曖昧性を減少させるための努力を怠ってきたためであり，上述の (2) から (4) の手順を繰り返すことは，労力を要するが，特に重要であると我々は考える。

現在，上述のサイクルを 1 回行い，新たに判明した問題点を検討しているところである。本稿では，今までの試みの成果と新たに判明した問題点を紹介し，その解決方法を検討する。

2 これまでの変更点

まず，1 回目のサイクルで変更した点を簡単に述べる [7]。連体修飾句の係り受けの曖昧性は抑え，連用修飾句の係り受けの曖昧性は残すことを基本方針としているが，大きな変更点は以下の 4 点である。

活用形の考慮 動詞句等が連体修飾句となるか連用修飾句となるかという区別に，動詞等の活用形の情報が重要である。活用形の情報が非終端記号にないと，構文解析における曖昧性を無意味に増大させることになる。

必須格情報の取り扱い 各動詞等の必須格情報は，一見すると，助詞句の係り先を制限することに有効であると考えられる。ところが，二格など表層的な情報だけでは判断が困難な格を構文解析で考慮することは無意味に

[1] 未変更のコーパスから抽出した文法でも構文解析の後に意味的な情報を利用して再解析することは可能だが，それならば構文解析の段階で解決できない構造まで厳密に決定しようとする文法規則は不要である。逆に，構文解析では解決できない曖昧性を出すことは構文解析で解決できる問題との判別を困難にする要因となり，その後の意味解析にも影響を与えると我々は考える。

[2] 以降，本稿では文脈自由文法を単に文法という。

表 1: 構文解析結果

	文法規則数	平均構文解析木数
変更前	1762	4.093×10^{12}
変更後	2740	2.183×10^{5}

曖昧性を増大させる要因となる。そこで，表層的な情報でほぼ決定可能なガ格とヲ格以外は無視する。

複合名詞内の構造 複合名詞内の構造を構文解析で解決することは不可能である。そこで，この構造を意味に関係なく右下がりの構造に制限する。

連体修飾句の係り受けの構造 連体修飾句の係り受け構造の曖昧性を動詞等のように助詞の情報で解消することは不可能である。そこで，複合名詞内の構造と同様，意味に関係なく右下がりの構造に制限する。

以上の点を中心に，EDR コーパス中の 9721 文に人手で構文構造を付けた文に対して，構文構造付きコーパス作成ツール [8] で変更を施した。そして，全文から文法を抽出し，全文を MSLR パーザ [10] で解析を行うと[3]，変更前のコーパス中の 3 文が解析途中でメモリオーバーフローが発生したため，この 3 文を除いた 9718 文で実験を行った。1 文あたりの形態素数は最小で 5，最大で 63，平均で 20.16 である。文法を 9718 文から抽出し，同じ 9718 文を構文解析すると，文法規則数が増えるが構文解析結果の曖昧性は大幅に抑えられることが分かる（表 1）。

さらに，確率モデルとして確率一般化 LR モデル（PGLR モデル）[3] を使用し，10-fold cross validation（8747 文で訓練，971 文で評価）を行った。ただし，解析結果の評価基準として文の正解率を以下のように定義する。

$$\text{文の正解率} = \frac{\text{出力した解析結果中に正解が含まれる文の数}}{\text{解析した文の総数}} \quad (1)$$

各順位以内における文の正解率を図 1 に示す。これより，変更前のコーパスから抽出した文法で上位 100 位の解析結果について意味情報を使用して再解析することと，変更後のコーパスから抽出した文法で上位 10 位の解析結果について再解析することがほぼ同じであることが分かる[4]。

参考として，PGLR モデルによる生成確率が 1 位の解析木について，文節の係り受けの正解率を調べた。ただし，本研究で扱う文法は句構造文法であるため，解析木の句構造を文節に区切り直し，右下がりの構造に制限している連体修飾句の係り受けは，すべて直後の名詞に係ることとした。これらの変更は人手によるところもあり，9718 文すべてを調べることはできなかったが，971 文を調べたところ 796 文 (81.98%) は文節の区切りがすべて正解であった。そのうちの 100 文を調べたところ 63 文は文中のすべての係り受けが正しく，文末の 2 文節を除く全 515 文節中 459 文節 (89.13%) の係り受けが正しかった[5]。これは，構文解析後の意味処理におけるベースラインとなる。

[3] MSLR パーザは形態素解析と構文解析を同時に行えるが，今回は入力を品詞列とすることで構文解析のみを行う。

[4] 変更前の文法で 90%を超えるには，上位 300 位くらいをとる必要がある [7]。

[5] 係り受け解析に関する他の研究の結果と比較すると，品詞列を入力として構文解析を行っているだけにも関わらず差が小さい。複雑な文ほど文節区切りが誤る可能性が高く，今回調べた 100 文には複雑な文が少なかったことがこれだけ高い係り受けの精度が得られた要因のひとつであると考える。

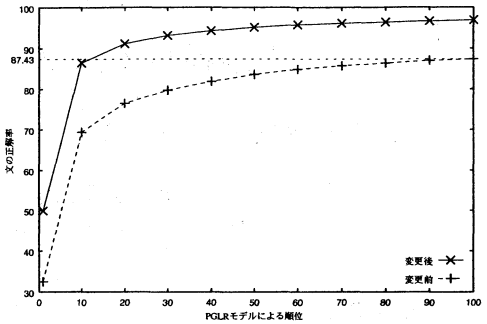

図 1: 各順位以内における文の正解率

3 新たな問題点

前節で示したように，コーパスを変更し文法を再抽出することで，構文解析結果の曖昧性を大幅に抑え，効率的に構文解析が行える。しかし，まだ無意味に曖昧性を出しているところがあり，さらに効率的な構文解析を行うためにはそれを解決する必要がある。

一般に，係り受け解析において並列構造を含む文の解析精度は，含まない文に比べて低くなる。そこで，我々が変更した文法についても，並列構造を含む文と含まない文で文の正解率にどの程度の差があるかを調べた（図 2）。ただし，このコーパスでは並列構造として並列名詞句，並列動詞句（形容詞句と判定詞句を含む）[6]，並列助詞句がそれぞれ 2069 文，178 文，160 文あり，それらのうち少なくともひとつを含む文が 2339 文ある。1 位のみの正解率を比較すると，並列構造を含む文の正解率は含まない文の正解率の半分程度しかないことがわかる。しかも，並列動詞句の場合は約 3%であり，ほとんど正解にならない。これまでは並列構造についての変更は一切していないが，上位 100 位までで比較すると，並列構造を含まない文の正解率は 99%近くであるのに対し，並列構造を含む文では 90%を少し上回る程度であることを考えると，解析精度をさらに向上させるためには並列構造について何らかの変更が必要である。

3.1 並列名詞句

「AのBとC」という並列名詞句の構造を考える（A，B，Cは名詞とする）。これまでの変更では並列名詞句の構造はそのままであり，この並列名詞句の構造は，「Aの」が「B」に係る構造と「BとC」に係る構造の 2 通りの曖昧性が出る。しかし，この構造の曖昧性は，意味的な情報を使わない限り解決できず，この曖昧性を残すことは曖昧性を無意味に増大させる要因となる。そこで，この曖昧性を構文解析結果として出さないように変更する必要がある。

ここで，「BとC」という最も単純な並列名詞句を考える。現在，この構造は各形態素をフラットな構造でまとめている（つまり，3 分木になっている）。しかし，助詞「と」を「の」

[6] 以降，並列形容詞句と並列判定詞句を含めて並列動詞句という。

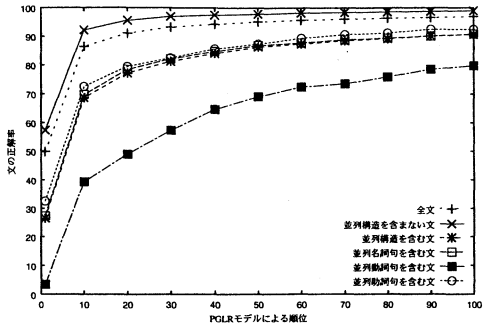

図 2: 並列構造を含む文と含まない文の正解率

図 3: 並列名詞句の構造の変更

に置き換えた場合（「BのC」）は，「B」と「の」がまとまって連体修飾句となり，それと「C」がまとまって名詞句となる。そこで，並列名詞句の場合でもこれと同じ構造，すなわち，「B」と「と」を先にまとめてから「C」とまとめる構造にすることで，並列名詞句の構造を一般的な連体修飾句の係り受けの構造と同じ扱いができるようになる（図 3）。これにより，「AのBとC」は「Aの」と「Bの」と「C」が右下がりの構造に制限され，構文解析結果に曖昧性は出なくなる。これは，「B，C」のように読点である場合，「BそしてC」のように接続詞である場合，「BあるいはC」のように副詞である場合も同様である。

しかし，次のような文がある。

1. 太郎は東京，花子は大阪 に住んでいる
2. 太郎は東京，花子は大阪 のアパートに住んでいる

どちらの例も，現在の文法では下線部分を先にまとめてから並列名詞句としているが，(1) の場合の係り受け構造を考えてみると，「太郎は」，「東京，」，「花子は」，「大阪に」はすべて「住んでいる」に係っていると考えることができる。つまり，構文解析の段階ではすべてを連用修飾句とし，後で連用修飾句どうしの関係を判定する際に並列関係であるかどうかを調べることにするべきである[7]。一方，(2) の場合は，「太郎は」と「花子は」は住んでいるに係るが，「東京，」と「大阪の」は「アパート」に係るべきである。これは係り受け関係が交差しているため，それぞれを別個に係り先とまとめるという (1) のような構造にはできない。係り受け関係が交差している場合の構造は現在検討中であり，当面は対象から除外する。

[7]「太郎が東京，花子が大阪に住んでいる」の場合，この方法では「太郎が」と「花子が」のふたつのガ格が別個に「住んでいる」に係る。これまでの文法では二重ガ格，二重ヲ格を認めていないため，この文は受理できなくなる。そこで，必須格情報はすべて無視し，二重ガ格，二重ヲ格を認める文法にすることで受理できるようにしている。予備実験では，必須格情報を完全に無視するとガ，ヲ格のみ考慮する場合より構文解析木数が約 100 倍になるが [7]，連体修飾句の係り受けの曖昧性は抑え，連用修飾句の係り受けの曖昧性を残すという基本方針に従うならばやむを得ないと考える。

3.2 並列動詞句

並列動詞句を含む文の正解率は非常に低い。この原因として訓練データとなるものが少ないこともあるが，並列関係であるかどうかの判定が特に困難であるとも考えられる。次の文を考える。

1. 歌ったり 踊ったり する
2. 歌を歌い，踊りを踊る
3. 7時に起き，8時に家を出る
4. 雨が降っていたので 外出しなかった

この中で (4) は前半部分が後半部分の理由にあたり，並列関係ではないと判断することは比較的容易である。しかし，(1)，(2)，(3) は人間でも判断が分かれるところであり，並列関係の定義をはっきり決めないとコーパス中でも一貫性を保持できなくなる（実際，コーパスを見てみると類似した文であるにもかかわらず並列と判断されている文と違う文が混在している）。ところで，ふたつの動詞句があるとき，その間の関係は並列関係か否かだけではない。(3) は時間的な前後関係を表し，(4) は理由を表している。さらに，「雨が降れば中止する」のように仮定を表すものなど様々な関係があるが，これらのうち並列関係だけを構文解析で決定しようとすること自体がナンセンスであり，動詞句間の関係の判定は，すべて構文解析後の意味処理で行うべきである。すなわち，並列動詞句の構造を構文解析結果として出さず，すべて連用修飾句と同じ構造にするべきである。

3.3 並列助詞句

並列助詞句は並列動詞句と同様，訓練データとなる文が少ないが，文の正解率は並列名詞句を含む文の場合とほぼ同じである。これは，並列関係にある助詞句を構成する助詞が同じであることが多く，また，異なる場合でも特殊なパターンとして扱えることがほとんどであり，助詞の情報を利用することで判定することができるからである。しかし，それでも並列構造を含まない場合に比べて正解率ははるかに低く，並列名詞句や並列動詞句と同様，構造を見直す必要がある。そこで，以下の例を考える。

1. 国政段階でも 個別産業レベルでも 影響力は小さい
2. 太郎は東京に 花子は大阪に 住んでいる
3. 東京から 大阪まで 新幹線で行く
4. 東京から 大阪まで の距離を調べる

(1)，(2)，(3) は，並列名詞句の場合で述べた「太郎は東京，花子は大阪に住んでいる」の例と同様，すべての助詞句を別個に動詞句に係る構造にすればよい[8]。同様に考えると，(4) は「東京から」が「距離」に係る構造になるべきであるが，助詞句が連体修飾句の働きもすることとなり，曖昧性が増大する要因になる。

ここで，助詞句が連体修飾句として働く例を調べてみた。

1. 2年に 1度の改選期

[8]「太郎は東京，花子は大阪に住んでいる」の例は，「東京」の後に助詞「に」が省略されていると考えれば (2) の例と同じになる。

2. 駅を 中心に発展する
3. 親友と つまらないケンカが もとで絶交した
4. 私が 外相の頃
5. この作品の主役が ペンギンというところが絶妙であった

(1), (2), (3) は係り先の名詞の性質を利用すれば決定できるかもしれないが, (4), (5) は「外相」,「ペンギン」という語だけで決定することは不可能に近い。助詞句が連体修飾句として働く例を見てみると, このほとんどは係り先の語の直後に助詞「の」,「に」,「で」や「という」,「とすれば」のような用言を含む助詞相当句があるもの (「この作品の主役が ペンギンというところ」) であった。これより, 係り先の名詞で判断するよりもその名詞の直後の助詞で判断する方が効果的であると考える。具体的には, 助詞「の」は判定詞「だ」の連体形, 助詞「に」と「で」は判定詞「だ」の連用形と考えることで「1度の」,「中心に」,「もとで」,「外相の」が判定詞句とみなし, それに助詞句が係るという構造とする。これは, それぞれの文が以下のように言い換えられることと関係がある。

1. 改選期は2年に一度だ
2. 発展するのは駅が中心だ
3. 親友と絶交したのはつまらないケンカがもとだ
4. その頃は私が外相だった

「東京から大阪までの距離を調べる」の例は,「東京から」が「大阪までの」に係る構造にすることになり, 並列助詞句の構造をなくすことができる。

4 おわりに

多様な言語現象を扱える大規模な文法を開発するためには構文構造付きコーパスからボトムアップに抽出するべきであるが, 構文解析の曖昧性を無意味に増大させるなど問題が多く, 実用に供されていないのが現状である。しかし, その最大の要因は曖昧性を抑えるための文法やコーパスの変更が不十分である点にあり, 曖昧性を増大させる要因を分析し, 文法やコーパスを変更することを繰り返すことによって, 構文解析のための実用的な大規模文法を構築できると我々は考えている。

この考えに基づき, 構文解析では解決できない構造の曖昧性を抑えた文法を使用して構文解析を行い, その後, 意味的な情報を利用して厳密な解析を行うというアプローチを採用した。そして, 具体的にどの曖昧性を抑えるべきかを検討し, コーパスを変更し, 文法を再抽出したが, それでもまだ並列構造を含む文の正解率が低いということが判明した。そこで, 本稿ではその並列構造の曖昧性を抑えるための構造の変更方法を提案した。現在, この方針に基づいてコーパスの構文構造を変更中であり, 正解率を算出するのに十分な量がないために実験を行っていないが, 正解率は大幅に上がると予想している。

今後の課題を以下に示す。

- 本稿で提案した並列構造の変更方法では, 並列構造を他の構造と同じにすることで曖昧性を抑えている。すなわち, 並列構造であるかどうかの判定を構文解析後の意味処理に先送りしていることになるが, これを再解析する手法を, 複合名詞内の構造や連体修飾句の係り受け構造の再解析も含めて考える必要がある。
- 現在のコーパスの変更では基本的に形態素レベル (品詞レベル) の変更は考慮していない。しかし, 構文解析における曖昧性を抑えるためには形態素区切りの基準や品詞体系の見直しが必要である。
- コーパスを作成する際に重要となるのは, それまでに作成したデータの管理である。作成方針に問題点が見つかり, 作成途中で方針が変更になった場合, それまでに作成したデータのうち変更が必要なものがすぐに分かれば, コーパス作成者にとって有用である。そのためにコーパスをデータベース化し, 該当するデータを検索するシステムを構築することを考えている。

参考文献

[1] James F. Allen, Donna K. Byron, Myroslava Dzikovska, George Ferguson, Lucian Galescu, and Amanda Stent. Toward conversational human-computer interaction. *AI Magazine*, Vol. 22, No. 4, pp. 27–37, 2001.

[2] Eugene Charniak. Tree-bank grammars. In *the 13th National Conference on Artificial Intelligence*, pp. 1031–1036, 1996.

[3] Kentaro Inui, Virach Sornlertlamvanich, Hozumi Tanaka, and Takenobu Tokunaga. Probabilistic GLR parsing: A new formalization and its impact on parsing performance. 自然言語処理, Vol. 5, No. 3, pp. 33–52, 1998.

[4] 井佐原均, 田中穂積. 日本語埋め込み文の構文解析における諸問題. 情報処理学会自然言語処理研究会, Vol. NL26, No. 4, 1981.

[5] Karen Jensen and Jean-Louis Binot. Disambiguating prepositional phrase attachments by using on-line dictionary definitions. *Computational Linguistics*, Vol. 13, No. 3–4, pp. 251–260, 1987.

[6] Yoshihiko Nitta, Atushi Okajima, Hiroyuki Kaji, Youichi Hidano, and Koichiro Ishihara. A proper treatment of syntax and semantics in machine translation. In *COLING 84*, pp. 159–166, 1984.

[7] 野呂智哉, 白井清昭, 徳永健伸, 田中穂積. 大規模日本語文法の開発 — 事例研究. 情報処理学会自然言語処理研究会, Vol. NL150, No. 22, pp. 149–156, 2002.

[8] 岡崎篤, 白井清昭, 徳永健伸, 田中穂積. 正しい構文木の選択を支援する構文木付きコーパス作成ツール. 人工知能学会 第15回全国大会, 2001.

[9] 白井清昭, 徳永健伸, 田中穂積. 括弧付きコーパスからの日本語確率文脈自由文法の自動抽出. 自然言語処理, Vol. 4, No. 1, pp. 125–146, 1997.

[10] 白井清昭, 植木正裕, 橋本泰一, 徳永健伸, 田中穂積. 自然言語解析のための MSLR パーザ・ツールキット. 自然言語処理, Vol. 7, No. 5, pp. 93–112, 2000.