社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

大規模日本語文法の開発 — 事例研究

野呂　智哉†　　白井　清昭††　　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　〒 152-8552 東京都目黒区大岡山 2-12-1
†† 北陸先端科学技術大学院大学 情報科学研究科　〒 923-1292 石川県能美郡辰口町旭台 1-1
E-mail: †{noro,take,tanaka}@cl.cs.titech.ac.jp, ††kshirai@jaist.ac.jp

あらまし　構文解析において，多様な言語現象を扱うためには大規模な文法が必要となるが，人手でトップダウンに作成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，コーパスと略す）があればそこから大規模な文法を抽出し，ボトムアップに文法を開発することができる。ところが，我々の経験によればコーパスから抽出した文法による構文解析は非常に多くの解析結果（曖昧性）を作り出し，それが解析精度の悪化や解析時間，使用メモリ量の増大の原因となる。そこで，無意味に曖昧性を増大させる要因を分析し，曖昧性を極力抑えるよう文法やコーパスを変更する必要がある。本論文では，意味的な情報を利用しなければ解決できない構造の決定は構文解析後の処理に任せ，構文解析で作り出される無意味な曖昧性を抑えた文法を開発するための具体的な方法を示すとともに，その有効性を実験的に明らかにしている。それにより，曖昧性を極力抑えた実用的な大規模日本語文法をボトムアップに開発することが十分可能であるとの見通しを得ている。

キーワード　大規模文脈自由文法，構文構造付きコーパス，構文解析

Building a Large-Scale Japanese Grammar — Case Study

Tomoya NORO†, Kiyoaki SHIRAI††, Takenobu TOKUNAGA†, and Hozumi TANAKA†

† Graduate School of Information Science and Engineering, Tokyo Institute of Technology
2-12-1 Ō-okayama, Meguro-ku, Tokyo, 152-8552 JAPAN
†† Graduate School of Information Science, Japan Advanced Institute of Science and Technology
1-1 Asahidai, Tatsunokuchi, Ishikawa, 923-1292 JAPAN
E-mail: †{noro,take,tanaka}@cl.cs.titech.ac.jp, ††kshirai@jaist.ac.jp

Abstract　A large-scale grammar is needed in parsing a variety of sentences, but it is difficult to build it manually. On the other hand, it is possible to build a large-scale grammar by deriving it from a large-scale tree parsed corpus(hereinafter, only abbriviated to a corpus). But because a great number of parse trees are created by using a grammar derived from a corpus, parsing accuracy gets worse, and to make matters worse, it takes a long time and a large amount of memory to parse sentences. For the above reasons, the corpus and the grammar should be modified to decrease the number of parse trees. This paper proposes the method to build a large-scale context free grammar that creates smaller number of parse trees by leaving the analysis of the structures that cannot be solved if no semantic information is considered, and shows the effectiveness of the grammar experimentally. We expect that it is possible to build a practical large-scale grammar that decreases the number of parse trees by using our method.

Key words　Large-Scale context free Grammars, Tree Parsed Corpora, Syntactic Analysis

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容易に開発できるが，分析可能な表現が限られ，網羅的でなく実用的でない。多様な言語現象を扱うためには文法の規模を大きくしなければならないが，大規模な文法を人手でトップダウンに開発することは非常に困難であることが知られている。一方，大規模な構文構造付きコーパス（以下，単に「コーパス」とした場合は構文構造付きコーパスを指す）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の文法を使用し，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良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 コーパスのように構文構造が付与されたコーパスが少ない。白井ら [11] は，構文ラベルが付与されていない括弧付きコーパス（EDR コーパス）(注1)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。このような手法で獲得した文法規則には，自動的に推測した構文ラベルに不自然なものが含まれるため，人間にとって理解しがたい文法体系となり，適切なものであるとは言えない。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，第 2 節で考察するように，人手で構築したコーパスには誤りが含まれることが避けられないという問題がある。そのため，そこから抽出した文法にはコーパス作成者の意図しない解析結果を生む規則が含まれてしまうことがある。それが曖昧性を無意味に増大させ，構文解析の精度の悪化や解析時間，使用メモリ量の増大の原因となる。Charniak はコーパスでの出現頻度の低い文法規則を削除するなどして解析精度の向上を図っているが，曖昧性を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ないことが問題である。

第二に，意味的な情報を使用しない限り構文解析では解決できない構造を解析結果として出力することも構文解析結果の曖昧性を増大させる要因となる。例えば，英語の場合，PP attachment 問題において，前置詞句の係り先の曖昧性の数は前置詞句の数に対する Catalan 数となることが知られている [3], [8]。その後の意味解析において，最終的にはこの多数の構文解析結果の中から正しい解析木をひとつ選び出すことになり，この作業が非常に困難になることが容易に予測される。意味的な情報を使用しない限り解決できない構造を構文解析結果の中に含めようとする試みは，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と我々は考えている。このような場合には，むしろ構文解析における無駄な曖昧性を極力抑えられるよう第 3 節で示す基準に従ってコーパスを変更し，そこから抽出した文法を使用して構文解析を行うことで無駄な曖昧性を抑え，意味的な情報を用いなければ決定できない構造の厳密な解析は，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。[6], [7], [9] (注2)。

以上のことから，ボトムアップに文法を開発する手順は以下のようになる。

（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やす文法規則の分析
（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に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。

本論文は，人手で構文木を付与したコーパスを出発点とし，そこから抽出した文法の問題点を分析してその具体的な解決方法を示し，その有効性を実験的に明らかにする一つの事例研究である。

2. 大規模なコーパス，文法の問題点

多様な言語現象を扱う大規模な文法を人手でトップダウンに作成することは困難であり，構文木付きの大規模なコーパスから抽出してボトムアップに開発すべきである。しかし，構文木付きのコーパスは人手で作成するため，後述するが，そこから抽出した文法を使用して構文解析すると曖昧性が増大するという問題が生まれる。曖昧性の増大は，解析精度の低下だけでなく，計算時間や使用メモリ量の増加につながるため，極力抑える必要がある。曖昧性が増大する要因には，以下の 4 種類がある。

（1） コーパス作成者が言語学的に誤った構造を付与すると，その構文木から抽出した文法規則が曖昧性を増大させる要因となることがある。

（例）「開発される」は終止・連体形であるにも関わらず，連用句になっている（図 1）。この構文木から抽出される文法規則を使用して生成される解析木は言語学的に誤ったものであり，無意味な曖昧性である(注3)。

（2） 同じ構造を付けるべきパターンに対して複数通りの構造を付与し，一貫性が取れていないと，そのパターンに対して構文解析した結果の数はその分だけ増大する。

（例）判定詞が直前の名詞句とどの段階で結合するか，2 通りの構造がコーパスに存在すると，判定詞を含む文の構文解析結

(注1)：括弧付けにより各構成素の依存関係を与えたコーパス

(注2)：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，その後の意味的な情報を利用した処理にも影響を与えると我々は考える。

(注3)：この例は要因 (3) で示している例と連動しているため，両者の要因を同時に解決しなければ，曖昧性を抑えることはできない。

果としてこの 2 通りが含まれる（図 2）(注4)。

（3） 構文解析時にコーパス作成者の意図しない解析木を生成する規則が存在すると，曖昧性が増大する。

（例）動詞句は連用節にも連体句にもなれる。コーパス作成者は形態素の情報を見ながら構造を付与するが，この構文木から文法を抽出すると形態素の情報が失われるため，動詞句「10 年ほど前に体をこわし」が連体句として「長男夫婦」を修飾する構造も構文解析の結果として生成される（図 3）。

（4） 意味的な情報を利用しない構文解析では解決できない問題は，統計情報を利用しても完全には解決できないので，構文解析の段階では無駄な曖昧性として残る。

（例）複合名詞の内部の構造は，それを構成する名詞等の意味が分からなければ決定不可能であり，構文解析では曖昧性が解消できないので，意味解析の段階で排除される無意味な曖昧性を多数生み出す文法規則の体系は好ましくない（図 4）。

構文解析における曖昧性を抑えた文法を開発するためには，以上に述べた問題点を解決しなければならない。特に，構文解析のための文法を開発する場合，(4) で示した構文解析では解決できない問題は，その後の意味的な解釈を前提として構文解析の段階には一通りの構造が得られるような文法体系にしておき，厳密な解析はその後の意味的な情報を含めた処理で行うべきである。

次節では，コーパスや文法の具体的な変更基準を示し，その変更の際に現れる問題点とその解決方法を述べる。

3. コーパス，文法の変更

第 2 節で述べた理由で，人手で作成した構文構造付き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，膨大な曖昧性を生み出し，性能が悪化する。例えば，EDR コーパスに人手で構文構造を付与した 9888 文から抽出した文法を使用して EDR コーパス中の 1000 文の品詞列（平均 20.50 形態素）を構文解析すると，平均で 4.85×10^{13} 個もの解析木が生成される。曖昧性を増大させる要因のうち，要因 (1) と (2) はコーパス作成者の単純なミスであり，発見され次第修正すべきであるが，要因 (3) と (4) は作成者のミスとは言えず，構文解析における無駄な曖昧性を極力抑えられるようその部分の文法体系を変更しなければならない。本節では，要因 (3)，(4) による曖昧性の増大を抑えるための文法体系（及び，構文構造）の具体的な変更方法について述べる。

3.1 活用形の考慮

理由 (3) の例で示したように，コーパス作成者は形態素情報を考慮して動詞句等が連用修飾語句になるか，連体修飾語句になるかを判断し，構文構造を付けるが，その際，非終端記号に区別がないと，その構文構造から単純に抽出した文法規則は形態素情報を考慮しないものとなり，動詞が連用形であるにも関わらずそれを含む動詞句が連体句になるという構造が構文解析の結果として生成される。この問題を解決するには，動詞等の

(注4)：「連邦政府指定の」が連用句となっていることが言語学的に誤っていると考えることもできる。

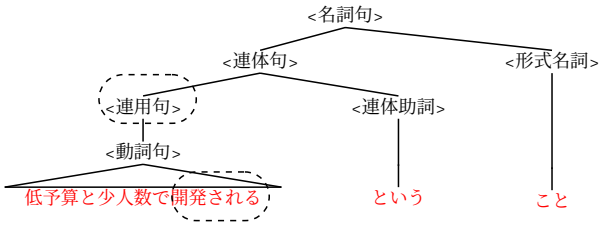
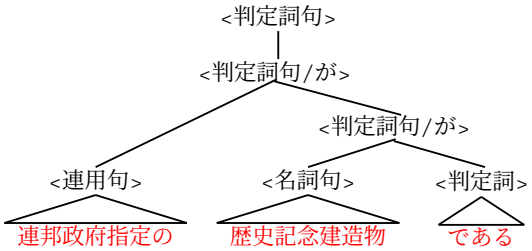


図 1 言語学的に誤った構造



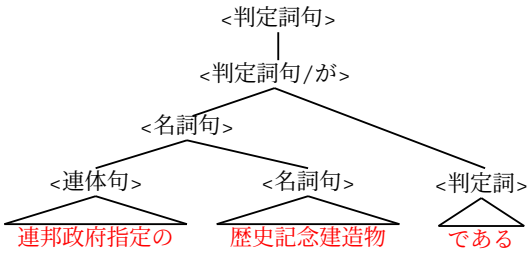


図 2 一貫性のない構造

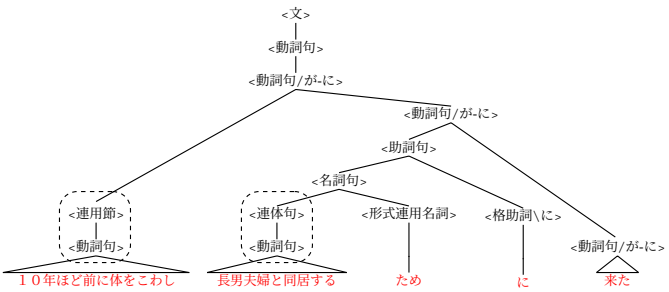


図 3 意図しない構造

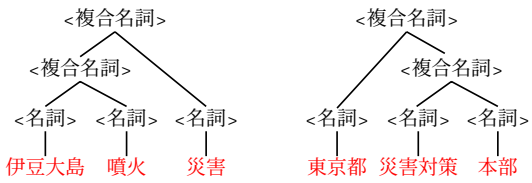


図 4 構文解析では解決できない問題

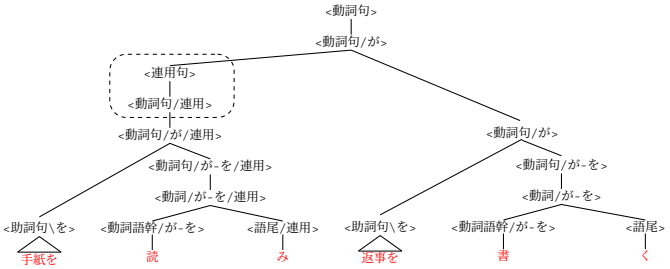


図 5　活用形を考慮した非終端記号

活用形の情報を非終端記号に反映させる必要があるが,

$$< \text{動詞} > \rightarrow < \text{動詞語幹} >< \text{語尾} > \qquad (1)$$

という規則について，すべての活用形を考慮した規則に変更すると，以下のようになる(注5)。

$$< \text{動詞/未然} > \rightarrow < \text{動詞語幹} >< \text{語尾/未然} > \qquad (2)$$

$$< \text{動詞/連用} > \rightarrow < \text{動詞語幹} >< \text{語尾/連用} > \qquad (3)$$

$$< \text{動詞/終止} > \rightarrow < \text{動詞語幹} >< \text{語尾/終止} > \qquad (4)$$

$$< \text{動詞/連体} > \rightarrow < \text{動詞語幹} >< \text{語尾/連体} > \qquad (5)$$

$$< \text{動詞/仮定} > \rightarrow < \text{動詞語幹} >< \text{語尾/仮定} > \qquad (6)$$

$$< \text{動詞/命令} > \rightarrow < \text{動詞語幹} >< \text{語尾/命令} > \qquad (7)$$

ところが，すべての活用形をそのまま非終端記号に付与することは文法規則の急増を招く。本研究におけるコーパスと文法の変更の目的は構文解析における曖昧性を抑えることにあり，活用形を完全に付与することではないので，活用形をそのまま付与するのではなく，連用句になり得る動詞句となり得ない動詞句を区別することで曖昧性を抑える（図 5）。こうすることで，< 動詞句/連用 > は連用句になれるが連体句にはなれず，逆に < 動詞句 > は連体句になれるが連用句になれない，とすることが可能となる。

3.2　複合名詞内の構造

理由 (4) の例で示したように，複合名詞内の構造は構文解析では解決できない問題であり，この曖昧性を構文解析の段階で出すことは無意味である。また，EDR コーパス中の 9888 文に含まれる名詞，複合名詞の数を調べると，複合名詞の数は平均で 1〜2 個，1 複合名詞を構成する形態素数も平均で 2〜3 個と少ないが（表 1），形態素解析の段階の曖昧性も考慮すると漢字 1 字で名詞となる場合もあり，結果として構文解析結果の中に含まれる複合名詞の数は非常に多くなり，さらに，1 複合名詞を構成する形態素数も多くなるため，曖昧性が急増する要因となる。以上の理由から，複合名詞内の構造はすべて右下がりの構造に制限し，厳密な解析は意味解析に先送りすべきである。

ところが，意味に関係なく構造を右下がりに制限することには問題がある。例えば,「実験/用/機械」のように接尾語「用」が入る場合，単純に右下がりにすると以下の規則ができる。

(注5)：動詞の場合，終止形と連体形は同型であるため，両方を合わせて < 動詞/終止・連体 >→< 動詞語幹 >< 語尾/終止・連体 > とすることも可能である。

表 1　1 文中に含まれる名詞

1 文あたり		1 複合名詞あたり	
名詞数	複合名詞数	形態素数	文字数
4.52 個	1.14 個	2.34 形態素	9.35 文字

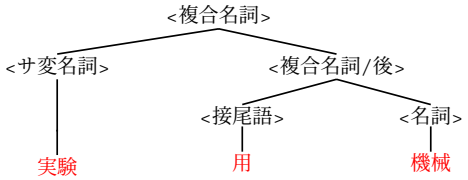


図 6　複合名詞内の構造

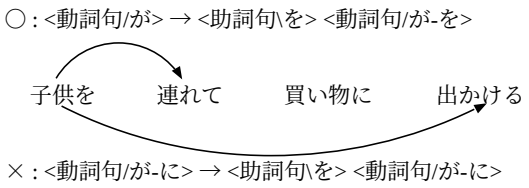


図 7　助詞句の係り先の決定

$$< \text{複合名詞} \rightarrow < \text{接尾語} >< \text{名詞} > \qquad (8)$$

この規則は，先頭が接尾語である複合名詞を認めるものであり，無意味な曖昧性を出す要因となる。そこで，図 6 のように複合名詞を構成する部分木の最上位の非終端記号とそれ以外の非終端記号を分けることにより，複合名詞の先頭になり得ない品詞が先頭になることを許す文法規則ができないようにする。

3.3　格情報の取り扱い

複合名詞内の構造以外に，構文解析の段階では解決が難しいものとして連用修飾語句の係り受けの関係を決定する問題があるが，動詞等の必須格情報を利用すれば，構文解析の段階でも解決可能な場合がある。例えば,「子供を連れて買い物に出かける」という文で，助詞句「子供を」は「連れて」,「出かける」のどちらに係るか曖昧であるが,「連れる」はヲ格を必須格とし,「出かける」は必須格としないことから，この曖昧性は構文解析で解決できる（図 7）。

しかし,「1 時に太郎が次郎に会う」という文では,「会う」はニ格を必須格とする動詞であるが,「1 時に」と「次郎に」のどちらが必須格に相当するかを構文解析の段階で解決することはできない。このように表層的な情報だけでは解決できない問題では，必須格情報を利用することが曖昧性の増大を招く。そこで，助詞の情報の有無による曖昧性の増減を調べるため，EDR コーパス 3133 文に対して以下の 5 通りの方法で構文構造を付与し，そこから抽出した文法を使用して同じ 3133 文を構文解析した。

（1）　意味を考慮した上で，すべての必須格情報を利用した場合

（2）　(1) に対し，動詞等の必須格情報，助詞句の格情報をすべて除去した場合

（3）　(1) に対し，動詞等の必須格情報だけすべて除去した場合

（4） (1) に対し，動詞等の必須格情報のうち，ガ格，ヲ格以外を除去した場合

（5） (4) に対し，格助詞と副助詞はすべて助詞の情報を非終端記号に付与した場合

構文解析の結果を表 2 に示す。(2)，(3)，(4) は逆に曖昧性が増大しているが，(5) ではわずかであるが減少している。(2) で曖昧性が急増している原因は，「１週間に３回 行く」のように助詞句が名詞等（この場合は数量詞）と結合する場合，助詞の情報を除いた規則では助詞に関係なく名詞等に結合できることになるためである。つまり，「太郎が次郎 に会う」で，「太郎が」が「次郎」と結合することを認めることとなり，曖昧性が増大する。ガ格とヲ格のみを必須格情報として利用することで曖昧性が抑えられるのは，ガ格とヲ格は表層的な情報でほぼ決定可能であることを示している。さらに，(5) で曖昧性が現象しているのは，「国政段階でも個別産業レベルでも 影響力は小さい」のように助詞句が並列となる例では，助詞句を構成する助詞が同じであるか，異なっていても「東京から大阪まで の距離」のような特殊なパターンであり，動詞等の必須格情報とは別に格助詞や副助詞の情報を利用することで曖昧性を抑えられることを示している。以上より，動詞等の必須格情報として利用するのはガ格とヲ格のみとし，その他の格助詞，副助詞の情報も動詞等の必須格情報とは別で利用することで曖昧性を抑え，それで解決できない連用修飾語句の係り先の曖昧性はそのまま残すこととする。

3.4 連体修飾語句の係り受けの構造

連用修飾語句の係り受けの構造の決定と同様，連体修飾語句の係り受けの構造も構文解析の段階では決定できない。連体修飾語句は助詞句のように助詞の情報を利用して曖昧性を抑えることはできず，複合名詞内の構造と同様に右下がりの構造に制限すべきである。

ところが，構造を右下がりに制限する際，問題点がふたつある。ひとつは，連体修飾語句の係り受けの構造の曖昧性には以下の 2 種類があることである。

- 連用修飾語句の範囲を変えないもの

（1） [[新しい 環境] への適応能力] がある

（2） [新しい[環境への適応能力]] がある

- 連用修飾語句の範囲を変えるもの

（1） [[１００年の 歴史] を持つ祭り]

（2） [１００年の[歴史を持つ祭り]]

前者は，連体修飾語句「新しい」が「環境」を修飾する場合でも，「適応能力」を修飾する場合でも，動詞「ある」を連用修飾先とする助詞句の範囲が「新しい環境への適応能力が」であることに違いはないが，後者の場合，連体修飾語句「１００年の」が「歴史」を修飾する場合と「祭り」を修飾する場合では，動詞「持つ」を連用修飾先とする助詞句の範囲が「１００年の歴史を」と「歴史を」となり，連体修飾語句の係り先によって範囲が変わる。連用修飾語句の係り受けの構造は曖昧性を残すこととしているので，連用修飾語句の範囲を変えるものについても連体修飾語句の係り受けの構造を制限すべきではない。そこで，連用修飾語句の範囲を変えない前者の場合は (2) の構造に制限し，範囲を変える後者の場合の曖昧性はそのまま残す。

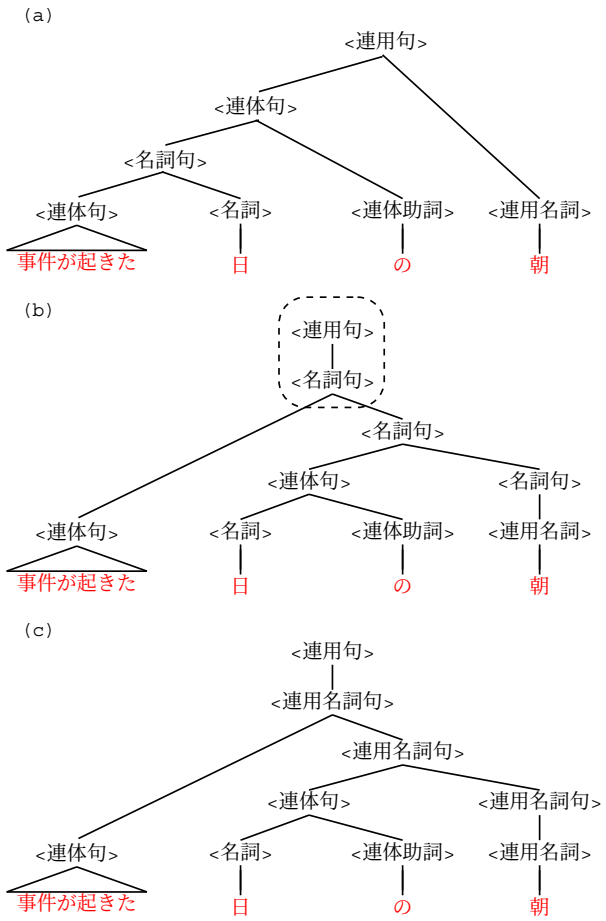


図 8 「事件が起きた日の朝」の構造

もうひとつの問題点は，単純に構造を右下がりに制限するだけでは表現できなくなる構造があることである。例えば「事件が起きた日の朝，彼に会った」という文では「事件が起きた日の朝」は連用修飾語句となるため，意味を考慮すると「事件が起きた日の」という連体句が連用名詞「朝」に係って連用句となる（図 8(a)）が，右下がりにすると名詞句が連用句になるという文法規則が必要となる（図 8(b)）。ところが，この規則を認めると，すべての名詞句が連用修飾可能となり，曖昧性の増大の要因となる。連用名詞は助詞を伴わずに連用修飾語句となり得る性質を持つので，構造を右下がりに制限する際には＜連用名詞句＞のように名詞句をさらに細かく分類し，＜連用名詞句＞は名詞句にも連用句にもなれるとすることで曖昧性の増大を抑えたまま構造を右下がりに制限することができる。

4. 評価実験

第 3 節で述べた基準に従い，EDR コーパス中の 2500 文に人手で構文構造を付けたコーパスに対して構文構造付きコーパス作成ツール [10] で変更を施した。そのうち，変更基準に合わない 51 文(注6)を除いた 2449 文（1 文あたりの平均形態素数 20.42

(注6)：今回は構文解析における曖昧性のみを考察の対象とし，コーパス変更の際，形態素レベルの誤り（品詞の割り当てや形態素区切りの誤り）の修正は行わず，それが原因で構文構造を付けられない文は除外した。

表 2　助詞の情報の有無による曖昧性の変化

パターン	(1)	(2)	(3)	(4)	(5)
構文木数	4.00×10^{10}	1.26×10^{14}	1.31×10^{12}	4.64×10^{10}	1.62×10^{10}

表 3　構文解析結果

		文法規則数	平均解析木数	解析失敗
(1)	変更前	1312	1.35×10^{11}	0
	変更後	1709	3.79×10^{4}	0
(2)	変更前	1247.6	1.04×10^{11}	125
	変更後	1602.4	2.29×10^{4}	218

個）について，変更前と変更後のコーパスからそれぞれ文法を抽出し，MSLR パーザ [12] で解析を行った。ただし，MSLR パーザは形態素解析と構文解析を同時に行うものであるが，入力は品詞列とすることで構文解析のみを行った。確率モデルとして確率一般化 LR モデル（PGLR モデル）[5] を使用し，以下の 2 通りの実験を行った。

（1） 文法は 2449 文全部から抽出し，PGLR モデルの学習には 2449 文を 5 組に分けた（1 組は 489 文，残りは 490 文）うちの 4 組を利用する。

（2） 文法抽出，PGLR モデルの学習の両方を同じ 4 組で行う。

評価は残りの 1 組を利用し，学習データを変えながら 5 回実験を繰り返し，その平均を取った。

抽出した文法規則数と構文解析結果として出力される解析木の数を表 3 に示す。これより，今回の変更によって構文解析の曖昧性が大幅に抑えられているが，実験 (2) では文法規則の不足による解析失敗の可能性が高くなっている。これは，構文解析における曖昧性を抑えるために非終端記号の数を増やし，その結果，文法規則数も増加しているので，文法抽出に使用するデータと評価に使用するデータを完全に分けると文法規則が不足するためであると考えられる。

次に，PGLR モデルによる生成確率が上位 10 位以内から 100 位以内の解析木について，文の正解率を求めた。文の正解率は以下のように定義する。

$$\text{文の正解率} = \frac{\text{出力した解析木の中に正しい木が含まれる文の数}}{\text{解析した文の総数}}$$

ここで「正しい木」とは，コーパスの構文構造と完全に一致する解析木を指し，「解析した文の総数」とは，解析に失敗した文を含めた総数である。従来の研究では，評価尺度として，括弧付けの再現率や適合率など部分的な構造の正しさを示す尺度を使用することが多い。しかし，例えば機械翻訳の場合，1 文全体の構文構造のうち，たとえ 1ヶ所でも構造の誤りがあれば，その文の翻訳結果はまったく異なるものになる。つまり，どれだけ部分な構造が正しいかではなく，1 文全体の構造が正しい解析結果がどれだけ得られるかが重要である。したがって，今回の実験で使用する評価尺度は，構造が完全に一致する文の割合を示すものとしている。

結果を図 9 に示す。この結果より，実験 (1)，(2) ともに変更後の方が正解率が高くなっているが，実験 (2) の方が差が小さくなっている。これは，文法規則の不足による解析失敗の割合が多いからであり，コーパスをさらに増やして解析に失敗する文の数が変更前と同程度になれば，実験 (1) に類似した結果が得られると考える。ちなみに，白井ら [11] の手法では，上位 30 位以内で 29.06%であったと報告している(注7)。

この実験で，注意すべきことがある。本研究の目的は構文解析のための大規模な文法を開発することにあり，意味的な情報を利用しない限り解決できない曖昧性は極力抑えられるようにコーパスや文法を変更している。ところが，変更前のコーパスや文法は構文解析の段階では解決できない問題の曖昧性を含んだものである。公平に比較するためには，変更前の文法を使用した解析結果について，意味的な情報を使わない限り解決できない構造をまとめてリランキングするか，もしくは，変更後の文法を使用した解析結果について，さらに意味的な情報を利用して厳密な解析をするかしなければならない。しかし，厳密な解析をした後に 1.35×10^{11} 個もの解析木をリランキングすることは二度手間である上に，曖昧性の増大は解析時間やメモリ使用量の増加にもつながる。今回は入力を品詞列とすることで構文解析の曖昧性のみを扱っているが，入力を平文として形態素解析の曖昧性も含めると，メモリ不足によって解析に失敗する可能性が高くなる。解析途中で枝刈りをする方法もあるが，構文解析の段階で解決できない曖昧性を含んだ状態で枝刈りをすると，正しい解析木を排除してしまい，完全性が損なわれる可能性が高くなる。予め構文解析で解決できない曖昧性を抑えた文法を使えば，解析途中で枝刈りをすることで正しい解析木を排除してしまう可能性は，変更前の文法を使用した場合よりも低くなるはずである。

さらに，実験 (1) について，変更後のコーパスから抽出した文法による文の正解率と同程度の正解率を得るためには変更前のコーパスから抽出した文法ではどのくらいの数の解析木を分析する必要があるかを調べるため，変更前の場合について上位 100 位から 500 位までの解析木も調べた。その結果を図 10 に示す。これより，約 90%の正解率を得るためには，変更後の文法では上位 30 位くらいを取ればよいのに対し，変更前では上位 300 位くらいまで取らなければならないことがわかる。

5. 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

(注7)：学習データとして約 18 万文を使用し，また，解析木中のすべての括弧付けが矛盾していない木を「正しい木」としているので，単純な比較はできないが，今後，コーパスの量を増やした場合の参考になると考える。

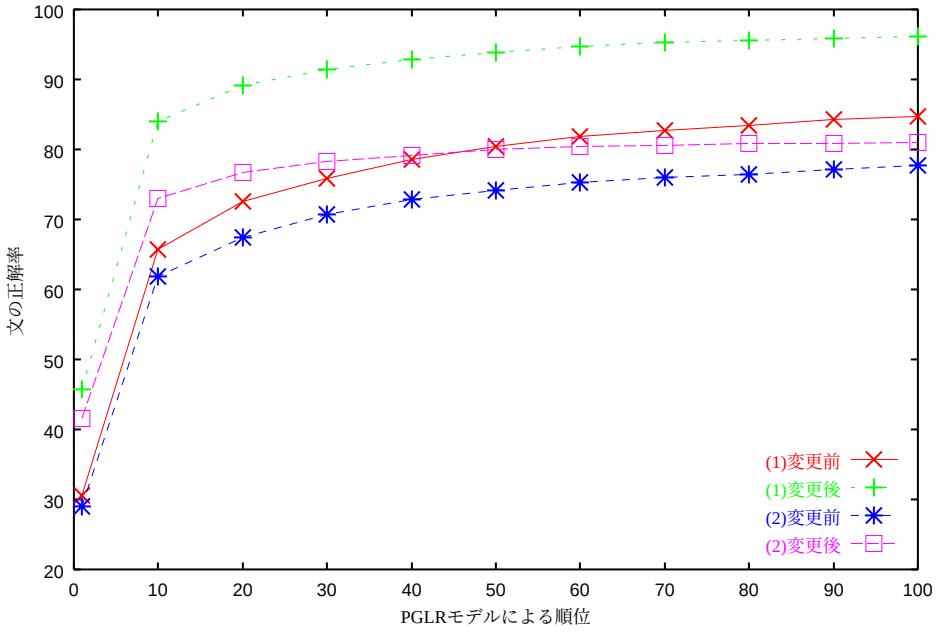


図 9　各順位以内における文の正解率

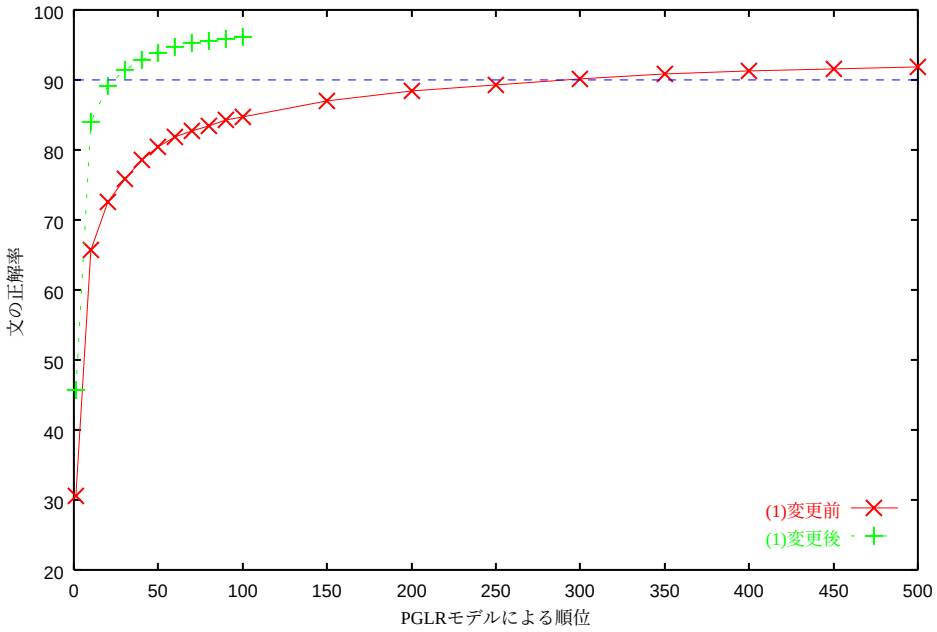


図 10　各順位以内における文の正解率

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。

本研究では，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，構文解析のための大規模日本語文法をボトムアップに構築するために，人手で作成したコーパスから抽出した文法の問題点とその具体的な解決方法を示した。また，それに基づいてコーパスを変更し，そこから抽出した文法で構文解析を行った結果，PGLR モデルの訓練用データと評価用データの両方から抽出した文法を使用した場合は，曖昧性が大幅に減るだけでなく，正しい解析木が生成確率順位の上位に入ることを明らかにしている。訓練用データのみから抽出した文法を使用した実験 (2) では，解析に失敗する文が多く，コーパスの変更による改善はあまり見られなかった。それは，時間と労力の問題から，現段階では約 2500 の構文木付きコーパスしか用意できなかったためだと思われる。実験 (1) で，解析に失敗する例が存在しない場合は改善の効果が大きいことから，さらにコーパスを増やして解析に失敗する文の数が変更前と同程度になれば変更前よりも結果が良くなると考えている。以上の結果から，大規模なコーパスから抽出した文法規則を，本論文で提案した指針に基づき変更することにより，大規模であるにも関わらず，構文解析結果の曖昧性を極力抑え，実用になる日本語文法を構築すること

が十分可能であるとの見通しを得た。

今後の課題を以下に示す。

- 現在，EDR コーパスに対して人手で構文木を付けたものが約 20000 文ある。これらすべてを本論文の考察結果に基づいて変更し，そこから抽出した文法を使用して実験を行い，この変更で解析精度が改善されることを示す必要がある。
- 構文解析結果の曖昧性を抑えた文法を抽出するためにコーパスの変更が必要となるが，この作業は多大な時間と労力を必要とする。無差別に選択した構文木から文法を抽出するのではなく，学習する価値のある構文木だけを選択して学習することで，学習に利用するコーパスの量を削減したり解析精度が向上したりできる [4] が，このような効率的な学習データの選択手法を考えることでコーパスの変更に要する時間や労力を削減できると考える。
- 訓練データのみから抽出した文法を使用すると，本論文で提案した変更では解析に失敗する例が増加し解析結果が改善しないが，これは，構文解析における無駄な曖昧性を抑えるために非終端記号数が増加し，それに伴い，文法規則数が増加したことが原因である。曖昧性の増大を抑えながら，文法規則数の増加も抑えられるような変更方法があれば，解析に失敗する例が減少し，その結果，解析精度が向上すると考えている。
- 複合名詞内の構造の曖昧性や連体修飾語句の係り先の曖昧性など，構文解析では解決できない問題ついては構造を制限することで厳密な解析を先送りしたが，意味的な情報等を利用して，これを再解析する手法を考える。
- 今回は品詞列を解析器の入力として構文解析における曖昧性のみを考慮したが，日本語のような膠着語では形態素解析の段階における形態素区切りの曖昧性も大きな問題となる。形態素解析の段階の曖昧性は文法の変更だけで抑えることは難しく，辞書の変更も考慮に入れる必要がある。また，形態素解析の段階の曖昧性を含めると使用メモリ量がさらに増大し，最後まで解析が終了しない可能性が高くなるが，解析途中で生成確率の低い解析木を破棄する枝刈りを行うことで，メモリ不足による解析失敗の可能性を抑えられる。この際，場合によっては正解となるべき解析木を途中で破棄してしまう可能性があるが，正解となる解析木が現れる順位（図 9）を考慮して枝刈りの基準を決定すれば，性能の低下を抑えることができると考える。

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] Kenneth Church and Ramesh Patil. Coping with syntactic ambiguity or how to put the block in the box on the table. *American Journal of Computational Linguistics*, Vol. 8, No. 3–4, pp. 139–149, 1982.

[4] Rebecca Hwa. Sample selection for statistical grammar induction. In *the 2000 joint SIGDAT Conference on EMNLP and VLC*, pp. 45–52, 2000.

[5] 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.

[6] 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 第 26 回自然言語処理研究会研究報告, 1981.

[7] 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.

[8] W.A.Martin, K.W.Church, and R.S.Patil. Preliminary analysis of a breadth-first parsing algorithm: Theoretical and experimental results. In Leonard Bolc, editor, *Natural Language Parsing Systems*, pp. 267–328. Springer-Verlag, 1987.

[9] 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.

[10] 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.

[11] 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.

[12] 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.