木構造に基づく決定的係り受け解析

北川 浩太郎　　田中 久美子
東京大学大学院 情報理工学系研究科
kitagawa@cl.ci.i.u-tokyo.ac.jp, kumiko@i.u-tokyo.ac.jp

1　はじめに

コーパスを利用して統計的な係り受け解析を行う手法は近年盛んに研究され、入力文に対して生成しうる係り受け木のなかから最も高いスコアのものを選択する最適化手法 [3][9]、局所的な係り関係を反復的に選択する決定的手法がそれぞれ提案されている。

決定的な手法は、Nivre による前方から順に係り関係を決定する手法 [10]、Yamada らによるボトムアップな解析手法 [12]、Goldberg らによる簡単な係り関係から順に決定する手法 [4] などが提案された。

本稿では既存の決定的手法に関し、解析動作に共通する操作に着目し、その問題点を考察した。係り関係を決定する順番には制約があり、誤りが伝播してしまう問題、複数の係り先候補を比較しなければ決定できない係り関係が存在する問題が挙げられる。これらの問題に対し、決定的な解析の中に探索を取り入れる Tree-Based モデル [6] を提案し、既存手法の拡張を試みた。

我々は既存の手法に対し、実際に Tree-Based モデルを用いて係り関係の探索範囲を拡大し、既存手法の問題点を解消した。評価実験においては、先に挙げた3つの手法に対して精度の向上を確認した。

2　係り受け解析

文 $x = (w_1, \ldots, w_n)$ における修飾・被修飾関係は、各単語を表す $V = \{1, \ldots, n\}$ をノード、$A \subseteq V \times V$ をエッジとする係り受けグラフによって表すことができる。本稿における係り受け解析とは、入力文に対し一般的に用いられる以下の制約 [11] の下で係り受けグラフを作成することを示す。

1. ROOT 以外の語は head を高々ひとつ持つ
2. 係り受けグラフは連結である
3. 係り受けグラフは非循環である
4. projective である (係り関係が交差しない)

コーパスと学習器を利用した統計的手法の研究は盛んに行われ、大きく分けて最適化手法と決定的手法の二つが存在する。それらの手法の位置づけを図1に表す。縦軸は最適な係り受けグラフを求める際に、他の生成可能なグラフをどれだけ探索しているのかを示しており、横軸は学習器を用いる際に、周辺の係り関係をどれだけ素性として利用できるかを示している。

最適化手法は全ての生成しうる係り受けグラフの中から最適なものを決定するものであり、探索範囲の広い手法として図1の上部に位置している。学習器を用いて単語間の係りやすさを求める際に、周辺の係り関係の構造情報を取り入れるアルゴリズムの研究が進められているが、Koo らの手法では計算量は $O(n^4)$ に至るなど、速度とのトレードオフが問題になっている。

一方、単語間の係り関係を一つずつ決定していくことで高速な解析を行う決定的手法についても数多く提案されており、探索範囲が限定的であるため、図1の下部に位置している。後に述べる Tree-Based モデルは、既存の決定的手法を拡張して探索範囲を広げるものである。

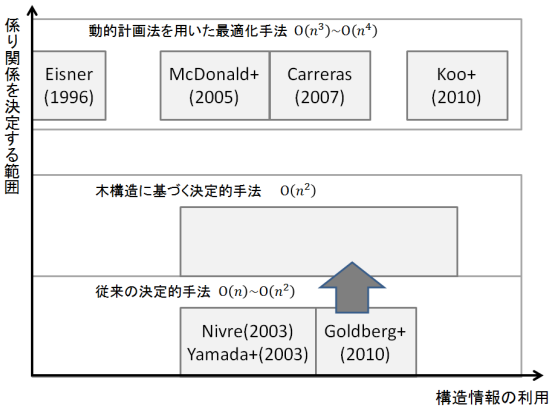

図 1: 係り受け解析手法

表 1: 既存手法の解析動作

	位置 i	操作 1	操作 2	操作 3	操作 4
Nivre	固定	$Right(i), Reduce(i), i \leftarrow i-1$	$Left(i+1), i \leftarrow i+1$	$Reduce(i), i \leftarrow i-1$	$i \leftarrow i+1$
Yamada	固定	$Right(i), Reduce(i)$	$Left(i+1), Reduce(i+1)$	$i \leftarrow i+1$[1]	
Goldberg	任意	$Right(i), Reduce(i)$	$Left(i), Reduce(i)$		

3 既存の決定的手法

3.1 共通する操作

既存の決定的手法は、単語列に対して異なる解析動作をそれぞれ反復的に選択するものであるが、個々の解析動作における具体的な操作内容の多くは共通している。入力された単語列に対して

- 隣接する単語間の係り関係の決定
- 係り先の決まった単語の消去

を行うことで係り受け木を生成している。隣接する単語との係り関係のみを調べることによって高速な解析を実現するとともに、既に係り先の決まった単語を取り除くことで、離れた単語間の係り関係も作成できるようになっている。

3.2 決定的解析手法

先に挙げた 3 つの決定的手法の解析動作の内容を表 1 にまとめた。Nivre の手法と Yamada の手法は、入力された単語列のある位置 i(初期値は 1) において、それぞれ 4 種類と 3 種類の解析動作を学習器を用いて選択する。一方、Goldberg の手法は任意の位置 i と二種類の解析動作のうち最も適切と考えられるものをを学習器で選択する。表 1 における操作 $Right(i)$ と $Left(i)$ は、i 番目に位置する単語の係り先を、右隣あるいは左隣に決定する操作であり、$Reduce(i)$ はすでに係り先の決まった i 番目の単語を単語列から除去する操作である。

図 2 は文 “A sign is the basic unit of language” を解析する途中の例だが、左から 3 番目に位置する単語 “unit” が左隣に係り、その後 “unit” を取り除いている。これは $Left(3)$ と $Reduce(3)$ の操作に相当する。

3.3 問題点 1: 解析順序の制約

係り受け解析の手法において、しばしば最適化手法と決定的手法は対比的に説明される。前者は局所的な素性を用いて大域的な解析を行い、後者は大域的な素性を用いて局所的な決定を行うというものである [11][13]。

[1]位置 i が文末であった場合, 文頭 $i = 0$ に戻る

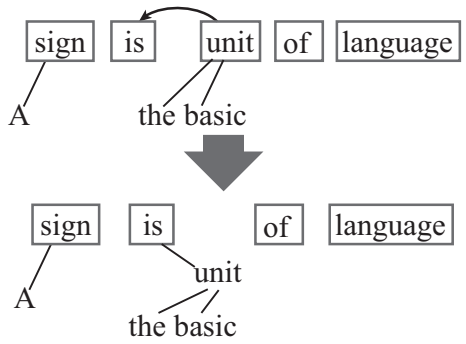

図 2: 決定的手法における係り先の決定と単語の除去

確かに、解析動作によって 1 つの係り関係を決定する際、その係り関係の周辺の素性を豊富に利用するように設計できる。しかし、図 2 で示された解析動作を行ったとき、決定されたのは “unit” と “is” の間の係り関係だけではない。“unit” を単語列から取り除いてしまうことによって、その後の解析において “of” と “unit” の係り関係を作成できなくなってしまう。このように、従来手法では係り関係の有無だけではなく、順番を考慮した解析を行わなくては正しい係り受け木を生成できないことになる。

そのような決定にはどれほど周辺の単語や情報を知る必要があるのかは難しい問題であり、順序に制約が存在することが解析を困難にしていると考えられる。

3.4 問題点 2: 探索範囲の局所性

もう一つの問題は探索範囲の狭さである。従来手法において $Left(i)$ や $Right(i)$ が決定するのはあくまでも隣接する単語間の局所的な係り関係の有無である。しかし、隣接する単語間の head-dependent 関係を決定する際に、i 番目の語の head 候補と比較していないため、相対的に w_i よりも head に相応しい単語を見逃す可能性が考えられる。

もちろん、解析動作を決定する際に学習器が利用できる素性の量は設計の工夫で増加させることはできる

表 2: Tree-Based モデルの解析動作

	位置 i	操作 1	操作 2	操作 3
Tree Nivre	固定	$Right(i), i \leftarrow i-1$	$Left(i+1)$	$i \leftarrow i+1$
Tree Yamada	固定	$Right(i)$	$Left(i+1)$	$i \leftarrow i+1$
Tree Goldberg	任意	$Right(i)$	$Left(i)$	

ものの、他の係り先候補との係り関係の有無を考慮することは困難である。

4 Tree-based モデル

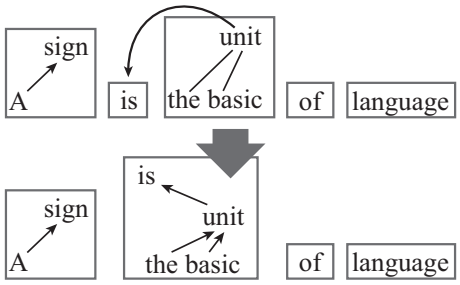

図 3: 提案手法における係り先の決定

上記の問題を解決するため、我々は Tree-Based モデルを提案した [6]。提案モデルは解析対象の文を構成する単位として、単語の代わりに木を用いたものである。木において head の決まっていない単語はその根に位置する語のみとなる。木 t_i, t_j 間の head-dependent 関係を決定するとは t_j の根にあたる語の head を t_i の中から決定することと定義し、係り関係が作成された場合には 2 つの木は結合されるものとする。

図 3 に Tree-Based モデルにおける解析例を示す。文は木から構成され、“unit” と “is” の間に、dependent-head の係り関係を作成した場合、“is”,“the”,“basic”,“unit” の 4 語からなる木が作られる。このとき、従来の決定的手法と異なり、単語は取り除かれずに木の内部に保存される。すると図 3 では、後に続く解析において “of” と “unit” の係り関係を作成可能となっている。これは第一の問題であった、解析順序の制約を取り除くものであり、正しい係り関係を決定する解析動作が原因となる解析誤りを防止している。

また、提案手法では隣接する木の間に存在する係り関係を考えるとき、head となり得る語の候補を探索し、最適な語を選択することによって、局所性の問題を一部解消している。

4.1 解析アルゴリズム

Tree-Based モデルで拡張された 3 つの手法における解析動作は表 2 である。単語を消すという操作はなくなり、隣接する木の間に係り関係を作る $Left(i)$,$Right(i)$ の操作によって係り受け木をつくる。係り受け木が生成されるまでの共通の流れを Algorithm 1 に示した。入力文を木の系列とみなし、反復的に解析動作を選択する。解析動作を行うにつれて木の数は減少していき、全体で一つの木が作成されたら終了する。

学習器を用いて最も高いスコアを与えた解析動作が選択されることになるが、学習器に用いる素性として、$Left(i)$ あるいは $Right(i)$ ならばどの単語に係るのかということを使用する。つまり、係り先となる候補は事前に決まっていなければスコアを求めることはできない。この係り先候補の選択については次節で述べる。

スコアを求める学習器を作るためには、あらかじめ係り関係の正解が与えられたコーパスを用いて、解析アルゴリズムを実行する。正解と異なる判定を学習器が与えた場合それを訓練事例とすることで、逐次学習することができる。

4.2 係り先候補の選択

上記アルゴリズムにおいて、木 t_i の根に位置する語の最も係り先らしい語を、隣接する木から選択する必要がある。ここでは、候補となる語を比較判定する二値分類器を使って繰り返し判定し、最終的な勝者を決定するトーナメントモデル [5] を用いた。

5 評価実験

提案手法の有効性を確かめるために Penn Treebank WSJ を用いた評価実験を行った。句構造から CoNLL-X Shared task[1] のフォーマットに変換し、学習データとして section2 から 21、テストデータとして section23 を使用した。学習機は訓練事例に対してパラメタをそのつど更新するオンライン学習の一つである Passive-Aggressive アルゴリズム [2] を Java で実装した。素

Algorithm 1 Tree-Based Model

```
//入力：文 s = w_1 ... w_n
//出力：係り関係を表すエッジ集合 (A)
//Acts: 解析動作の集合
A = {}
tree = t_1 ... t_n ← w_i ... w_n
while (length(tree) > 1) do
  // 最大のスコアを与える動作 best を選択
  best ← arg max_{act∈Acts} score(act.tournament)
  // その動作に基づいて係り関係を作成
  if ∃(head, dependent) ← edgeFor(best) then
    A.add( (head, dependent) )
  end if
end while
return A

//act のスコアを求める際には、隣接する木から
//係り先候補として最適な単語の選択を行う
```

性設計は [4] と同等のものを使用し、単語のユニグラム、バイグラム、前置詞の情報に加え、単語間の距離といった構造情報を用いた。

各単語について head となる語を正しく推測できた確率を単語正解率、正しい係り受け木を求められた文の割合を文正解率、一文あたりの解析時間の平均である解析速度を求めた。

表 3 に記した結果のように、各手法において精度の向上が確認された。探索を行うため、一文あたりにかかる計算量は増加するものの、平均解析時間それほど大きくは悪化しなかった。

表 3: 実験結果

	単語正解率	文正解率	時間 [ミリ秒/文]
Nivre	88.0%	31.7%	4.6
Tree Nivre	89.1%	32.6%	11.6
Yamada	87.6%	31.3%	9.7
Tree Yamada	89.0%	32.3%	10.3
Goldberg	90.8.%	38.3%	46.2
Tree Goldberg	91.2%	37.9%	65.3

6 まとめ

本稿では決定的な係り受け解析手法に探索をとりいれる Tree-Based モデルを提案し、既存の手法を拡張した。

従来手法には係り関係を決定する順序に制約があること、係り先となり得る複数の候補から相対的に良いものを選ぶことができないことといった共通する問題点を含んでいた。その原因は、従来手法における解析動作が、単語列に対して隣接する単語間の係り関係を決定したり、単語を除去するといった極めて局所的な操作から構成されることにある。

Tree-Based モデルは解析時に探索する範囲を広げ、木と木の係り関係を一つの解析動作で決定する。例に挙げた 3 つの既存手法全てに対し、複数の係り先候補を考慮した解析を実現させ、係り関係を決定する順番に設けられていた制約条件を除去した。評価実験においては、探索範囲が増加するため平均解析時間は遅くなるものの、解析精度が上昇することを確認した。

参考文献

[1] S. Buchholz, and E. Marsi, CoNLL-X shared task on multilingual dependency parsing,*Proceedings of CoNLL*, pp.149–164, 2006.

[2] K. Crammer, O. Dekel. J. Keshet, S. Shalev-Shwartz and Y. Singer, Online Passive-Aggressive Algorithms, *Journal of Machine Learning Research*, 2003.

[3] J. Eisner, Three new probabilistic models for dependency parsing: An exploration, *Proceedings of COLING*, pp.340–345, 1996.

[4] Y. Goldberg, and M. Elhadad, An Efficient Algorithm for Easy-First Non-Directional Dependency Parsing, *Proceedings of NAACL*, 2010

[5] M. Iwatate, M. Asahara, and Y. Matsumoto, Japanese dependency parsing using a tournament model, *Proceedings of COLING*, pp.361–368, 2008.

[6] K. Kitagawa, and K. Tanaka-Ishii, Tree-based deterministic dependency parsing – An applycation for Nivre's method –, *Proceedings of ACL*, 2010.

[7] T. Koo, and M. Collins, Efficient third-order dependency parsers, *Proceedings of ACL*, pp.1–11, 2010.

[8] T. Kudo and Y. Matsumoto, Japanese dependency structure analysis based on support vector machines, *Proceedings of EMNLP*, 2000.

[9] R. McDonald and F. Pereira, Online learning of approximate dependency parsing algorithms, *Proceedings of EACL*, pp.81–88, 2006.

[10] J. Nivre, An efficient algorithm for projective dependency parsing, *Proceedings of IWPT*, pp.149–160, 2003.

[11] J. Nivre, Algorithms for deterministic incremental dependency parsing, *Computational Linguistics*, vol.34,num.4, pp.513–553, 2008.

[12] H. Yamada and Y. Matsumoto, Statistical dependency analysis with support vector machines, *Proceedings of IWPT*, pp.195–206, 2003.

[13] Y. Zhang and S. Clark, A tale of two parsers: investigating and combining graph-based and transition-based dependency parsing using beam-search, *Proceedings of EMNLP pp.562–571, 2008.*