係り受け解析との同時実行に基づく日本語文の語順整序と読点挿入

宮地 航太 †1　大野 誠寛 †2　松原 茂樹 †1

†1 名古屋大学　†2 東京電機大学

miyachi.kota@j.mbox.nagoya-u.ac.jp

1 まえがき

日本語は語順が比較的自由であるため，語順を強く意識しなくても，意味の通じる文を書くことが出来る．しかし実際には，語順に関して選好が存在しているため，文法的には間違っていないものの読みにくい語順を持った文が作成されることがある．また読点についても同様に，打ち方は比較的自由であるが選好が存在するため，読みやすい文を作成するためには適切な位置に読点を打つ必要がある．例えば，以下の 2 つの文

文 1 私は家を都会に憧れ出た．

文 2 私は，都会に憧れ家を出た．

では，例文 1 はそのままでは読みにくいが，文 2 のように文節を並べ替え，読点を挿入すれば読みやすくなる [1].

文を読みやすい語順に整えるという語順整序は，推敲支援や文生成などに応用でき，これまでにも幾つか研究されている．大野ら [2] は，係り受け解析との同時実行による語順整序手法を提案している．また，日本語テキストに読点を挿入する手法として村田ら [3] は，文構造を明確にする，並列する語の区切りを示す，など，読点の用法ごとにその出現傾向を捉える特徴素を設定している．しかし，入力文が読みにくい語順であれば，係り受け解析や読点挿入の精度が低下し，また，係り受け解析の精度が低いと，語順整序や読点挿入の精度も低下するという問題がある．

そこで本論文では，推敲支援のための要素技術として，読みにくい語順をもった日本語文に対して，係り受け解析，語順整序，読点挿入を同時実行する手法を提案する．本手法は，係り受け構造が付与されていない文を入力とし，係り受け解析，語順整序，読点挿入を同時に行う．それらの同時実行により，入力文の語順だけでなく，整序後に生成する語順も考慮しながら，文の係り受けと読点位置を解析することができる．

2 日本語の語順・読点・係り受け

語順が係り受けと関連することをふまえ，先行研究 [4] では，事前に係り受け解析が施されることを前提に，その得られる構文情報を利用して語順整序を行っている．しかし，係り受け解析は一般に，新聞記事など読みやすい文に付与された係り受け構造から学習を行っているため，入力文が読みにくい語順であると精度が低下することになる．また，読点挿入についても同様に新聞記事などから学習を行っているため，入力文が読みにくい語順であると精度が低下する [5].

一方，係り受け解析の前に語順を整えると，係り受けの情報を利用できず，語順整序の精度は低下すると考えられる．また，係り受け解析では句読点の情報を利用するため，読点挿入の前に係り受け解析を行うと，係り受け解析の精度は低下すると考えられる．

このように係り受け解析，語順整序，読点挿入は互いに依存しているといえる．したがって，読みにくい語順を整え，かつ，読点を挿入するという問題に対するアプローチとしては，全てを同時に実行する手法が適しているといえる．

3 提案手法

本手法では，意味は伝わるものの読みにくい語順を持った文が入力されることを想定し，その文に対して，係り受け解析を行うと同時に，読みやすい語順と読点位置を同定する．すなわち本手法では，入力文に対して係り受け解析，語順整序，読点挿入を同時実行する．

これら 3 つの処理を同時実行するための戦略の一つとして，係り受け解析と語順整序の同時実行を実現した大野ら [2] の戦略を，読点挿入を含める形で拡張することが考えられる．すなわち，1 文全体に対して考えられる係り受け構造と語順，読点位置のあらゆるパターンの中から，最尤のパターンを動的計画法により探索するという戦略である．しかし，この戦略を本研究で採用しようとすると，係り受け構造と語順に加え，読点位置も含めたあらゆるパターンを考える必要があり，その数が膨大となる．また，各処理の非独立性から動的計画法を単純には適用できないため，最尤のパターンを効率的に探索することは容易ではない．

そこで本手法では，入力文中の局所的な 2 文節に着目して，それらの間の係り受け関係や語順，読点有無を同時的に決めることを繰り返すという戦略により，1 文に対する係り受け解析，語順整序，読点挿入の同時実行を実現する．本節では，まず 3.1 節で，入力文に対して係り受け構造，読みやすい語順と読点位置を決定するアルゴリズムについて述べる．次に，3.2 節では，アルゴリズムの中で操作選択を行う際に用いる確率モデルについて述べる．

3.1 アルゴリズム

本研究では，日本語係り受け解析を行う Shift-Reduce アルゴリズム [6] を拡張することにより，入力文中の局所的な 2 文節間の係り受け関係や語順，読点有無を同時的に決めることを繰り返すという戦略を実現する．

Shift-Reduce アルゴリズム [6] は，キューとスタックを用意し，スタック Top の文節がキュー Front の文節に係るか否かに応じて Shift と Reduce という 2 つの操作の選択を繰り返すことによって，1 文全体の係り受け構造を決定する．本研究では，このアルゴリズム [6] に対して，読点挿入を行うための操作（Shift-Comma, Reduce-Comma）と，語順変更を行うための操作（Swap）及び語順変更スタックの追加を主な拡張として行い，係り受け解析，語順整序，読点挿入の同時実行を実現した．なお以下では，語順変更スタックと区別するため，単にスタックと呼ぶものは Shift-Reduce アルゴリズム [6] において元々使用されるスタックを指すものとする．

3.1.1 基本動作

本手法により係り受け解析，語順整序，読点挿入を同時実行する様子を図 1 に示す．

本手法では，次の手順で入力文の文節列を先頭から順に処理する．

1. 入力文節列を入力の語順でキューに格納し，スタックと語順変更スタックを共に空とする．
2. 語順変更スタックが空であればスタック Top の文節とキュー Front の文節を，語順変更スタックが空でなければスタック Top の文節と語順変更スタック Top の文節を，それぞれ操作対象として，これらの間に対する操作（Shift, Shift-Comma, Reduce, Reduce-Comma, Swap のいずれか）を選択する．
3. 2 を繰り返し，キューと語順変更スタックが空となり，かつ，スタックに文末文節のみが積まれた状態になれば終了する．

ここで，各操作の内容は以下の通りである．なお以下では，操作対象の 2 文節のうち，スタック Top の文節を前方文節，もう一方を後方文節と呼ぶ．

Shift: 前方文節が後方文節に係らないことを決定し，後方文節をスタックに移す．

Shift-Comma: Shift の操作に加えて，前方文節と後方文節の間に読点挿入することを決定する．

Reduce: 前方文節が後方文節に係ることを決定し，前方文節をスタックから削除した上で，後方文節の子ノードとして追加し係り受け木を構成する．なお，ある状況においては特殊な動作を行うが，その詳細は 3.1.3 で述べる．

Reduce-Comma: Reduce の操作に加えて，前方文節と後方文節の間に読点挿入することを決定する．

Swap: 前方文節と後方文節の語順の入替を決定し，前方文節，後方文節の順に語順変更スタックにプッシュする．語順変更スタックを用意する理由は，語順変更が行われた場合に，操作対象の 2 文節が関わる係り受け解析や読点挿入の判断をリセットし，再度，判定し直すためである．なお，前方文節や後方文節が子ノードを持っている（すなわち，別の文節から係られると決定されている）場合は，その語順を保つ形で，全ての子孫を語順変更スタックにプッシュする．

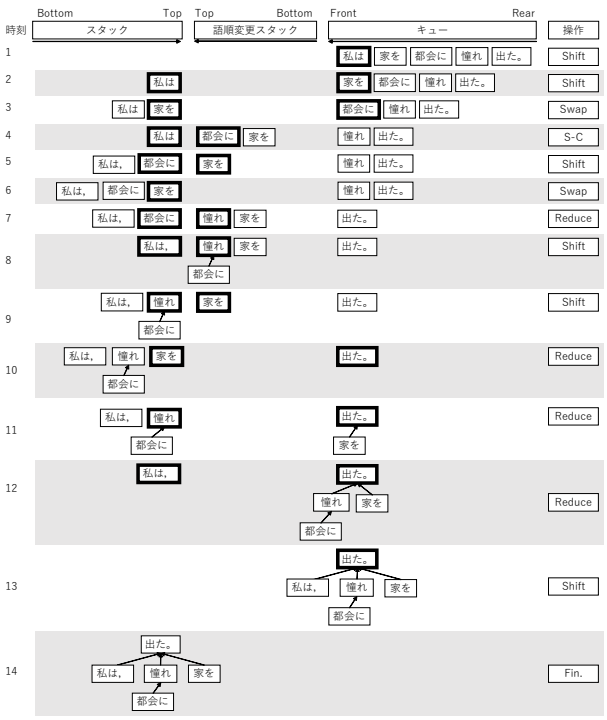

図 1: 提案手法の動作例

具体的な動作例を図 1 をもとに説明する．なお，図 1 では操作対象が太枠で示されている．初期状態の時刻 1 では，キューに入力文が格納され，スタックと語順変更スタックは共に空とするため，キュー Front のみが操作対象となり Shift が選択される．その結果,「私は」がスタックにプッシュされる．時刻 2 では,「私は」が「家を」に係らないとするため，Shift が選択され,「家を」がスタックにプッシュされる．時刻 3 では,「家を」と「都会に」の語順を入れ替えるため，Swap が選択され，この順で語順変更スタックにプッシュされる．時刻 4 では,「語順変更スタックが空でないので，私は」と「都会に」が操作対象となり，これらの間において，係り受け関係はなく，かつ，読点は打たれるとするため，Shift-Comma が選択される．その結果,「都会に」がスタックにプッシュされ,「私は」の後に読点が挿入される．飛んで時刻 7 では,「都会に」と「憧れ」の間に係り受け関係があるとするため，Reduce が選択される．その結果,「都会に」は，スタックから削除され,「憧れ」の子ノードとして追加される．最後に時刻 14 でキューと語順変更スタックが空となり，かつ，スタックに文末文節のみが積まれているため処理が終了する．

3.1.2 操作の制限

3.1.1 で示した手順 2 では 5 つの操作のいずれかを選択するとしているが，実際には，アルゴリズムの動作および日本語文法という二つの側面からの制約により，スタック，語順変更スタック，キューの状態に応じて，選択できる操作は制限される．以下では，どのような場合にどのような選択制限を行うのかを説明す

る．
[スタックが空である場合]：前方文節がない状況のため，Shift を必ず選択するようにする．なお，この状態は，初期状態や，文末文節以外の文節が全て Reduce された場合に発生する．
[スタックが空でない，かつ，語順変更スタックが空，かつ，キューの要素が残り一つ（入力語順における文末文節だけ）である場合]：Shift，Shift-Comma，Swap を操作の選択候補から外す．Shift，Shift-Comma を外す理由は，これらが選択されると，キューと語順変更スタックが共に空になり後方文節がない状況となり，その後の操作の動作が不可能になるためである．Swap を外す理由は，入力における文末文節の語順を変更させないためである．本研究では，読みにくい語順であるが意味は伝わる文が入力されることを想定しており，語順変更後だけでなく入力の語順においても係り受けの後方修飾性が満たされることを仮定している．そのため，入力における文末文節は出力においても必ず文末文節となる．
[前方文節が後方文節より入力の語順において後方にある場合]：Swap を操作の選択候補から外す．これは，既に Swap により変更した語順をもとに戻させないためである．これにより，同じ文節間で Swap が繰り返され無限ループに陥ることを回避できる．
[後方文節が係り受け木の子ノードを持つ（他の文節から係られると決定されている）場合]：Shift-Comma 及び Reduce-Comma を操作の選択候補から外す．これは，後方文節が係り受け木の子ノードを持つ場合，前方文節と隣接しているのは，後方文節をルートとする係り受け木の最左に位置する子孫ノードの文節であり，前方文節とその子孫ノードの文節との間に読点を打つか否かは既に決められているためである．

3.1.3 Reduce 及び Reduce-Comma の特殊動作

本研究では，読みにくい語順だが意味は伝わる文が入力されることを想定しており，語順変更後だけでなく入力の語順においても係り受けの非交差性及び後方修飾性が満たされることを仮定している．そのため，出力の語順だけでなく入力の語順においても，これらの制約を満たす係り受け構造を出力する必要がある．

そこで本手法では，Reduce が選択された際に，前方文節に加えてスタック上の複数の文節（以下，Reduce 対象）を係り元とし，後方文節を係り先とする係り受け関係を一括で決定する場合がある．具体的には，入力の語順及び現在の解析結果の語順において前方文節と後方文節の間に位置する文節がスタック上に存在する場合，それらと前方文節の全てを Reduce 対象として，一括して後方文節に係ると決定する．すなわち，Reduce 対象をスタックから削除した上で，後方文節の子ノードとして追加し係り受け木を構成する．なお，Reduce 対象の収集は，その中の文節を前方文節とみなして再帰的に行い，後方修飾性と非交差性を保つように，一括して Reduce すべき文節をスタック上で網羅的に収集する．

3.2 操作選択のための確率モデル

本節では，アルゴリズム中で操作を選択する際に用いる確率モデルについて述べる．

操作を機械的に選択する方法の一つとしては，正しい出力を得るための操作系列から，操作そのものを正解ラベルとして学習して分類モデルを構築することが考えられる．しかし，本研究で想定する「意味は伝わるものの読みにくい語順を持った文」と正解データのペアを大量に収集することは難しい．そこで本手法では，各操作を実施した後の解析結果の妥当性を確率モデルにより推定し，その値が最も高くなる操作を選択することとした．

以下では，時刻 t で選択される操作を f_t，時刻 t までの操作系列を $\boldsymbol{f}_t = f_1 f_2 \cdots f_t$ と表す．また，入力文の文節列を $B = b_1 b_2 \cdots b_n$（b_i は入力の語順における i 番目の文節）とし，入力文に対して時刻 t までの操作系列による解析結果を $S^{\boldsymbol{f}_t}$ とする．$S^{\boldsymbol{f}_t}$ は，時刻 t までの操作系列で決定された，語順 $O^{\boldsymbol{f}_t} = \{o^{\boldsymbol{f}_t}_{1,2}, o^{\boldsymbol{f}_t}_{2,3}, \cdots, o^{\boldsymbol{f}_t}_{1,n}, o^{\boldsymbol{f}_t}_{2,3}, \cdots, o^{\boldsymbol{f}_t}_{i,j}, \cdots, o^{\boldsymbol{f}_t}_{n-1,n}\}$，読点位置 $C^{\boldsymbol{f}_t} = \{c^{\boldsymbol{f}_t}_{1,2}, c^{\boldsymbol{f}_t}_{2,3}, \cdots, c^{\boldsymbol{f}_t}_{1,n}, c^{\boldsymbol{f}_t}_{2,3}, \cdots, c^{\boldsymbol{f}_t}_{i,j}, \cdots, c^{\boldsymbol{f}_t}_{n-1,n}\}$，係り受け構造 $D^{\boldsymbol{f}_t} = \{d^{\boldsymbol{f}_t}_{1,2}, d^{\boldsymbol{f}_t}_{2,3}, \cdots, d^{\boldsymbol{f}_t}_{1,n}, d^{\boldsymbol{f}_t}_{2,3}, \cdots, d^{\boldsymbol{f}_t}_{i,j}, \cdots, d^{\boldsymbol{f}_t}_{n-1,n}\}$ の三項組として定義され，$R^{\boldsymbol{f}_t} = \langle O^{\boldsymbol{f}_t}, C^{\boldsymbol{f}_t}, D^{\boldsymbol{f}_t} \rangle$ と書く．

ここで，$o^{\boldsymbol{f}_t}_{i,j} (1 \leq i < j \leq n)$ は，時刻 t の操作後における，文節 b_i と b_j の間の順序関係を表し，b_i が b_j より文頭側に位置するか（$o^{\boldsymbol{f}_t}_{i,j} = 1$），否か（$o^{\boldsymbol{f}_t}_{i,j} = 0$）の 2 値をとる．また，$c^{\boldsymbol{f}_t}_{i,j} (1 \leq i < j \leq n)$ は，時刻 t の操作後において，隣接している 2 文節 b_i と b_j の間に読点が有るか（$c^{\boldsymbol{f}_t}_{i,j} = 1$），無いか（$c^{\boldsymbol{f}_t}_{i,j} = 0$) のいずれかの値をとる．最後に，$d^{\boldsymbol{f}_t}_{i,j} (1 \leq i < j \leq n)$ は，時刻 t の操作後において，2 文節 b_i と b_j との間に係り受け関係があるか（$d^{\boldsymbol{f}_t}_{i,j} = 1$），否か（$d^{\boldsymbol{f}_t}_{i,j} = 0$）の 2 値をとる．これらの値は全て初期状態では 0 をとるとする．

3.2.1 確率モデル

本手法では時刻 t で選択する操作 f_t を式 (1) により求める．なお，時刻 t における前方文節を b_i，後方文節を b_j とし，$S^{\boldsymbol{f}_{t-1} f_t}$ と $S^{\boldsymbol{f}_{t-1}}$ の差分が $o^{\boldsymbol{f}_{t-1} f_t}_{i,j}, c^{\boldsymbol{f}_{t-1} f_t}_{i,j}, d^{\boldsymbol{f}_{t-1} f_t}_{i,j}$ であるとする[1]．

$$
\begin{aligned}
& argmax_{f_t} P(S^{\boldsymbol{f}_{t-1} f_t} | B, S^{\boldsymbol{f}_{t-1}}) \\
& \approx argmax_{f_t} P(o^{\boldsymbol{f}_{t-1} f_t}_{i,j}, c^{\boldsymbol{f}_{t-1} f_t}_{i,j}, d^{\boldsymbol{f}_{t-1} f_t}_{i,j} | B, S^{\boldsymbol{f}_{t-1}}) \\
& = argmax_{f_t} \Big(P(o^{\boldsymbol{f}_{t-1} f_t}_{i,j} | B, S^{\boldsymbol{f}_{t-1}}) \\
& \times P(c^{\boldsymbol{f}_{t-1} f_t}_{i,j} | B, S^{\boldsymbol{f}_{t-1}}, o^{\boldsymbol{f}_{t-1} f_t}_{i,j}) \\
& \times P(d^{\boldsymbol{f}_{t-1} f_t}_{i,j} | B, S^{\boldsymbol{f}_{t-1}}, o^{\boldsymbol{f}_{t-1} f_t}_{i,j}, c^{\boldsymbol{f}_{t-1} f_t}_{i,j}) \Big)
\end{aligned}
\tag{1}
$$

$P(o^{\boldsymbol{f}_{t-1} f_t}_{i,j} | B, S^{\boldsymbol{f}_{t-1}})$ は，文節列 B に対して，時刻 $t-1$ までの操作系列による解析結果 $S^{\boldsymbol{f}_{t-1}}$ が与えられたと

[1] 3.1.3 の特殊動作が起こる場合など，実際の差分は異なることがある．

き，文節 b_i と b_j の語順を $o_{i,j}^{\boldsymbol{f}_{t-1}f_t}$ とすることが妥当な確率を，$P(c_{i,j}^{\boldsymbol{f}_{t-1}}|B, S^{\boldsymbol{f}_{t-1}}, o_{i,j}^{\boldsymbol{f}_{t-1}f_t})$ は，文節列 B に対して，$S^{\boldsymbol{f}_{t-1}}$ と $o_{i,j}^{\boldsymbol{f}_{t-1}f_t}$ が与えられたとき，隣接している文節 b_i と b_j の間の読点位置を $c_{i,j}^{\boldsymbol{f}_{t-1}f_t}$ とすることが妥当な確率を，$P(d^{\boldsymbol{f}_{t-1}}|B, S^{\boldsymbol{f}_{t-1}}, o_{i,j}^{\boldsymbol{f}_{t-1}f_t}, c_{i,j}^{\boldsymbol{f}_{t-1}f_t})$ は，文節列 B に対して，$S^{\boldsymbol{f}_{t-1}}$ と $o_{i,j}^{\boldsymbol{f}_{t-1}f_t}, c_{i,j}^{\boldsymbol{f}_{t-1}f_t}$ が与えられたとき，文節 b_i と b_j の係り受け関係を $d_{i,j}^{\boldsymbol{f}_{t-1}f_t}$ とすることが妥当な確率を，それぞれ表す．これらの確率はいずれも機械学習手法により推定する．

3.2.2 機械学習で用いる素性

$P(o_{i,j}^{\boldsymbol{f}_{t-1}f_t}|B, S^{\boldsymbol{f}_{t-1}})$ を推定する際には，文献 [2] の素性のうち，語順を推定する際に用いられた，受け文節に関する素性を除く全ての素性を用いる．$P(c_{i,j}^{\boldsymbol{f}_{t-1}f_t}|B, S^{\boldsymbol{f}_{t-1}}, o_{i,j}^{\boldsymbol{f}_{t-1}f_t})$ を推定する際には，文献 [3] で用いられた素性のうち，前方文節の係り受け情報を使うことなく取得可能な素性を用いる．$P(d_{i,j}^{\boldsymbol{f}_{t-1}f_t}|B, S^{\boldsymbol{f}_{t-1}}, o_{i,j}^{\boldsymbol{f}_{t-1}f_t}, c_{i,j}^{\boldsymbol{f}_{t-1}f_t})$ を推定する際には，文献 [6] で用いられた全ての素性を用いる．

4 評価実験

読みにくい日本語文の語順整序および読点挿入における本手法の性能を評価するため，新聞記事を用いた実験を実施した．新聞記事中の文から擬似的に作成した読みにくい語順の文に対して本手法を適用し，元の文の語順と読点をどの程度再現できるかで評価した．

4.1 評価用データの作成

本研究では，新聞記事中の文は読みやすい語順で書かれていることを前提に，以下の手順で読みにくい語順の文データを 1000 文作成しテストデータとした．（1）読みにくい語順の文を新聞記事中から擬似的に作成する [2].（2）その語順において可能な限り読みやすくなるように人手で読点を付与する [5].

4.2 実験概要

式 (1) における各確率を推定するための機械学習モデルには勾配ブースティングマシン（GBM）を採用し，そのツールとして LightGBM[2] をデフォルトのまま使用した．学習には京大テキストコーパス Ver.4.0[7] に収録されているテキストから 4.1 節のデータ作成に用いた文を除く 35,404 文のデータを用いた．

語順整序の評価では，文献 [2] と同様に，文単位正解率（語順整序後の語順が元の文と完全に一致している文の割合）と 2 文節単位正解率（2 文節ずつ取り上げたときの文節の順序関係が元の文のそれと一致しているものの割合）を測定した．

読点挿入では，正解文と語順が完全一致している文のみを対象に評価を行った．評価方法は文献 [5] と同様に，京大コーパスの読点位置を正解の読点位置とし，正解に対する再現率，及び，適合率により行った．

[2]https://lightgbm.readthedocs.io/en/latest/

表 1: 実験結果

語順		読点		
2 文節単位	文単位	再現率	適合率	F 値
72.61% (24,209/33,343)	10.90% (109/1,000)	48.44% (31/64)	60.78% (31/51)	53.91

入力文：
勝敗を，危険を避けロングキックで争っている傾向が強い．
出力文：
危険を避け，ロングキックで勝敗を争っている傾向が強い．

図 2: 語順整序及び読点挿入の成功例

4.3 実験結果

提案手法の語順整序の正解率と，正解文と語順が完全一致した出力文における読点挿入の再現率，適合率，F 値を表 1 に示す．語順整序や読点挿入の性能は十分でないものの，図 2 に示すように，出力文と正解文で語順と読点位置が完全一致した例も 65 文存在していた．以上より，本手法の実現可能性を確認した．

5 まとめ

本論文では，Shift-Reduce アルゴリズムを拡張することによって，読みにくい語順の文に対して係り受け解析，語順整序，読点挿入を同時実行する手法を提案した．読みにくい語順の文データを用いた評価実験の結果，本手法の実現可能性を確認した．今後は，確率の推定に用いる素性や機械学習手法を見直すことにより，精度を向上させたい．

謝辞 本研究は，一部，科研費 No. 26280082，No. 16K00300 及び No. 19K12127 により実施した．

参考文献

[1] 日本語記述文法研究会. 現代日本語文法 7. くろしお出版, 2009.

[2] 大野誠寛, 吉田和史, 加藤芳秀, 松原茂樹. 係り受け解析との同時実行に基づく日本語文の語順整序. 信学論, Vol. J99-D, No. 2, pp. 201–213, 2016.

[3] 村田匡輝, 大野誠寛, 松原茂樹. 読点の用法的分類に基づく日本語テキストへの自動読点挿入. 信学論, Vol. J95-D, No. 9, pp. 1783–1793, 2012.

[4] 内元清貴, 村田真樹, 馬青, 関根聡, 井佐原均. コーパスからの語順の学習. 自然言語処理, Vol. 7, No. 4, pp. 163–180, 2000.

[5] 宮地航太, 大野誠寛, 松原茂樹. 読みにくい語順の文への読点の自動挿入. 言語処理学会第 25 回年次大会発表論文集, pp. 1308–1311, 2019.

[6] 颯々野学. 日本語係り受け解析の線形時間アルゴリズム. 自然言語処理, Vol. 14, No. 1, pp. 3–18, 2007.

[7] 黒橋禎夫, 長尾眞. 京都大学テキストコーパス・プロジェクト. 言語処理学会第 3 回年次大会論文集, pp. 115–118, 1997.