情報処理学会論文誌　数理モデル化と応用　Vol.5　No.3　95–106 (Sep. 2012)

Pitman-Yor過程に基づく確率的木挿入文法モデル

進藤 裕之[1,a)]　藤野 昭典[1]　永田 昌明[1]

受付日 2012年4月19日，再受付日 2012年6月7日，
採録日 2012年7月9日

概要：Pitman-Yor 過程に基づく木挿入文法の確率モデルを提案する．提案モデルとマルコフ連鎖モンテカルロ法を組み合わせることで，発見的な手法に頼ることなく，構文木コーパスから木挿入文法の統計的な学習が可能となる．木挿入文法は，部分木の挿入操作により少数の規則で様々な構文パターンを表現できる一方で，統計的な学習には高い計算コストを要する．そこで，挿入操作を含む文法モデルを等価な文脈自由文法へ変換し，効率的に確率モデルの学習ができることを示す．提案手法を用いて構文解析の実験を行ったところ，比較的少量の学習用データでは文脈自由文法や木置換文法と比較して解析精度の向上を実現した．また，学習用データの規模が大きいときは，提案モデルは木置換文法のモデルと比較して約20%コンパクトでありながら，ほぼ同等の構文解析結果を達成した．

キーワード：木挿入文法，Pitman-Yor 過程，構文解析

A Probabilistic Model of Tree Insertion Grammars Based on Pitman-Yor Processes

Hiroyuki Shindo[1,a)]　Akinori Fujino[1]　Masaaki Nagata[1]

Received: April 19, 2012, Revised: June 7, 2012,
Accepted: July 9, 2012

Abstract: We propose a probabilistic model of Tree Insertion Grammars (TIG) based on the Pitman-Yor process. Instead of relying on heuristic rules to extract grammars, our model and Markov Chain Monte Carlo (MCMC) method automatically learns TIG rules from treebank data. Tree insertion is helpful for modeling syntax patterns accurately with compact grammar rules, however, it suffers from high computational cost for statistical learning. In this paper, we propose an efficient method for learning TIG by mapping our model to context-free grammars (CFG). The experimental parsing results show that our model outperforms a standard CFG and Tree Substitution Grammars (TSG) for a small dataset. For a large dataset, our model obtains comparable results to the TSG, making the number of grammar rules approximately 20% smaller than with TSG.

Keywords: tree insertion grammars, Pitman-Yor process, parsing

1. はじめに

木挿入文法（Tree Insertion Grammars）[19] は，構文木を生成する形式文法モデルの 1 つであり，文法情報を利用した機械翻訳などの言語処理タスクへ適用されている [7], [8]．文脈自由文法が非終端記号の再帰的な書き換え（置換操作）により文の構造を導出するのに対して，木挿入文法では，部分木（subtree）の再帰的な書き換え（置換・挿入操作）により文の構造を導出する．したがって，木挿入文法は文脈自由文法の自然な拡張であるといえる．木挿入文法では，任意の大きさの部分木を文法規則として利用するため，たとえば述語項構造などの言語学的な特徴を単一の規則でとらえることができるという利点がある [18]．また，近年さかんに研究されている木置換文法（Tree Substitution Grammars）[4], [18] は，木挿入文法と同様に部分木の再帰

[1] NTT コミュニケーション科学基礎研究所
NTT Communication Science Laboratories, Souraku, Kyoto 619–0237, Japan
[a)] shindo.hiroyuki@lab.ntt.co.jp

的な書き換えに基づく文法モデルである．しかし，木置換文法は置換操作のみにより文の構造を導出するモデルであり，挿入操作を行うことができない．したがって，木置換文法は木挿入文法のサブセットである．

従来より，木挿入文法の規則を人手で記述したり，Penn Treebank [13] などの構文木コーパスから発見的手法を用いて自動獲得したりする研究が行われてきた [1], [2], [22]．発見的手法とは，たとえば構文木のノードが主辞であるか，または付加詞であるかなどの言語学的な知見に基づいて人手で作成されたルールであり，統計的な情報は考慮していない．木挿入文法の規則を人手で記述するには，言語学に関する高度な専門知識が必要となり，多大な時間や費用を要するという問題点がある．また，構文木コーパスから規則を獲得するための発見的手法は，言語やコーパスの仕様に依存するため汎用性が低く，データに過学習しやすいという問題点が指摘されている [18]．

これらの問題を解決するため，我々は木挿入文法の確率モデルを考案し，木挿入文法の規則を構文木コーパスから統計的に自動学習する手法を提案する．ただし，一般的な構文木コーパスには，木挿入文法の規則に関する情報は付与されていないため，教師なし学習によってそれらを獲得する．木挿入文法では任意の大きさの部分木を規則として扱うため，膨大な部分木の可能性を考慮しなければならない．

このような膨大な部分木の確率分布を扱うために，本研究ではノンパラメトリックベイズモデルの一種である Pitman-Yor 過程 [17] を用いる．具体的には，部分木を確率変数とする多項分布を考え，事前分布として Pitman-Yor 過程を導入した確率モデル化を行う．Pitman-Yor 過程は，ベキ則に従う確率分布を生成することができ，言語データのモデル化に有効であることが示されている [4]．また，Pitman-Yor 過程に基づく確率モデルの学習にマルコフ連鎖モンテカルロ（MCMC）法を用いることによって，膨大な部分木の全可能性をあらかじめ列挙することなく，データに応じて適応的に生成することができる．ただし，木挿入文法の確率モデルに対して従来研究と同様の MCMC 法を直接適用することは困難であるため，我々は木挿入文法分解という手法を新たに提案し，木挿入文法を等価な文脈自由文法に変換して効率的に確率モデルの学習が可能であることを示す．本手法は言語やコーパスに関する先見的知見を必要とせず，様々なデータへ適用可能である．

英語の構文木コーパスを用いて提案モデルの学習を行った結果，木挿入文法の挿入操作は全体の規則数を大幅に減らす効果があり，コンパクトな文法で様々な構文パターンを表現可能であった．また，提案モデルの学習によって獲得された木挿入文法の部分木を構文解析タスクへ適用した結果，構文木コーパスが少量のときは挿入操作が効果的に働き，従来の木置換文法に基づく手法よりも高精度を実現した．構文木コーパスが十分な量のときは，従来の木置換文法に基づく手法より約 20%少ない規則数で，ほぼ同等の精度を達成した．

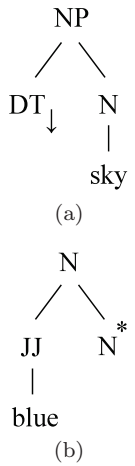

図 1　(a) 初期木の例．境界ノードは記号 “↓” で表している．(b) 補助木の例．末尾ノードは記号 “∗” で表している

Fig. 1　(a) An example of initial tree. (b) An example of auxiliary tree.

全体の構成は以下のとおりである．2 章では，木挿入文法に関する背景知識と関連研究について述べる．3 章では，Pitman-Yor 過程に基づく木挿入文法の確率モデルと，MCMC 法による効率的な学習のための木挿入文法分解法を新たに提案する．4 章では，英語の構文木コーパスを用いて本手法を適用した実験の結果を示し，最後に 5 章で結論を述べる．

2. 木挿入文法

木挿入文法は，5 タプル：$\mathcal{G} = (V_N, V_T, S, I, A)$ で表される．それぞれ，

- V_N：非終端記号（nonterminal symbol）の有限集合
- V_T：終端記号（terminal symbol）の有限集合
- $S \in V_N$：開始記号（start symbol）
- I：初期木（initial tree）の有限集合
- A：補助木（auxiliary tree）の有限集合

である．**図 1** に初期木と補助木の例を示す．図 1 において，“NP” は名詞句，“DT” は限定詞，“N” は名詞，“JJ” は形容詞を表す．また，図 1 (a), (b) はそれぞれ，“(NP (DT) (N sky))” や “(N (JJ blue) (N))” とも表記する．

非終端記号 V_N とは，形式文法の文法規則に使用される記号であり，たとえば図 1 (a) の “NP” や “DT” などである．非終端記号は，初期木または補助木のノードラベルとして用いられる．終端記号 V_T とは，形式文法で生成される文字列であり，たとえば図 1 (a) の “sky” である．したがって，終端記号は，初期木または補助木の葉ノードのラベルにしか現れない．

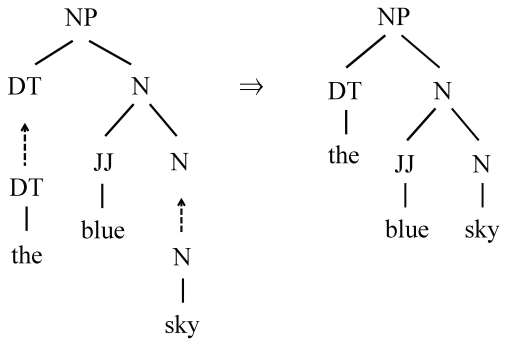

図 2　置換操作の例
Fig. 2　Example substitution operation.

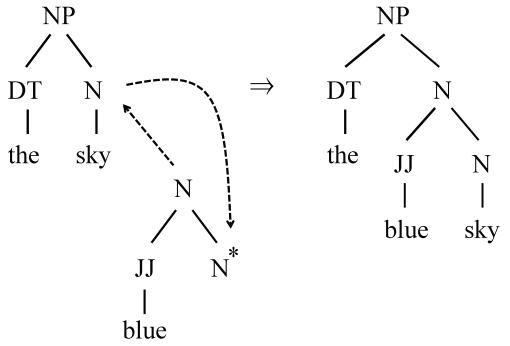

図 3　挿入操作の例
Fig. 3　Example insertion operation.

初期木は，各ノードに非終端記号または終端記号の付与された木構造である．初期木の葉ノードの中で非終端記号が付与されているノードは，特に境界ノード（frontier node）と呼ばれ，他の初期木または補助木によって書き換えられることを表す目印となる（図 1 (a) の記号 “↓”）．補助木は，初期木と同様に，各ノードに非終端記号または終端記号の付与された木構造であるが，根ノードと同じ非終端記号が付与されている特別な葉ノードが必ず 1 つ存在する．この葉ノードは末尾ノード（foot node）と呼ばれる（図 1 (b) の記号 “∗”）．末尾ノードは，補助木の挿入によって切り取られた部分木が結合するノードであることを示しており，詳細は後述する．初期木と補助木をまとめて基本木と呼び，基本木の各ノードの中で葉ノードでも根ノードでもないノードを内部ノード（internal node）と呼ぶ．

木挿入文法は，様々な基本木を組み合わせて文全体の構文木を生成するモデルである．木挿入文法の導出過程は，まず開始記号 S を根ノードとして，置換または挿入と呼ばれる 2 種類の操作によって基本木を結合していく．**図 2** および**図 3** に置換操作と挿入操作の例をそれぞれ示す．置換操作は，部分的に構築された構文木の境界ノードを初期木で置き換える操作である．ただし，境界ノードと，初期木の根ノードの非終端記号は同じでなければならない．一方，挿入操作は，部分的に構築された構文木の内部ノードに対して，補助木を挿入する操作である．ただし，内部ノードと，補助木の根ノードの非終端記号は同じでなければならない．図 3 で示されているように，補助木の挿入によって切り取られた内部ノード以下の部分木は，補助木の末尾ノードと結合する．本研究の木挿入文法モデルでは，図 1 (b) のような末尾ノードが根ノードの直接の子供となるような単純な補助木のみを仮定する．このような仮定を置かない場合，補助木の末尾ノードの探索に計算コストを要するため，末尾ノード探索のための効率的なアルゴリズムが別途必要となる．構文木データから初期木を推定するには，構文木に含まれる全ノードの中から，初期木の根ノードおよび葉ノードを決定すればよい．一方，補助木の推定には，根ノード，葉ノードに加えて末尾ノードの位置も決定する必要がある．本研究では，末尾ノードが根ノードの直接の子供となるような補助木のみを許容することにより，末尾ノードを探索する計算コストを省略し，初期木と補助木の学習を同じ探索アルゴリズムで扱う．上記の制約を仮定しない木挿入文法の確率モデル化および効率的な学習法の構築は，今度の課題である．

木挿入文法の挿入操作を禁止し，初期木の置換操作のみで構文木を生成するモデルを木置換文法と呼ぶ．近年，様々な木置換文法の確率モデルが提案され，構文木コーパスから統計的に部分木を学習することに成功している [4], [5], [18]．たとえば，Cohn ら [5] や Post ら [18] の研究では，Dirichlet 過程を用いて木置換文法の確率モデルを構築し，MCMC 法により部分木の学習を行う手法を提案している．また，Johnson ら [9] は，木置換文法の一種である Adaptor 文法を提案し，Pitman-Yor 過程を利用した確率モデル化を行っている．ただし，Adaptor 文法では，部分木のすべての葉ノードは必ず終端記号が付与されているという制約があり，一般的な木置換文法とは異なる．

我々の確率モデルは，木置換文法の確率モデルと比較して，挿入操作が含まれているという点が大きく異なる．挿入操作は，形容詞・副詞などの修飾語や記号類などの文中に挿入される語句を，コンパクトな部分木として表現できる．図 3 の例では，“the blue sky” だけでなく “the sky” も文法的に正しい．したがって，“blue” を挿入操作として表現することで，任意の “前置詞+名詞” 句の間へ挿入する汎用的な文法パターンとしてとらえることができる．一方，木置換文法では挿入操作が禁止されているため，“前置詞+形容詞+名詞” と “前置詞+名詞” を別々のパターンとして記憶する必要がある．

挿入操作は柔軟であるがゆえに，統計的な学習による部分木の獲得は木置換文法と比べて計算コストが高い．木置換文法では，部分木は必ず構文木の中に連続した領域として保存されているため，部分木の学習は構文木の分割問題として扱うことができる．一方，木挿入文法では，挿入操作によって 1 つの部分木が複数の断片となり構文木に埋め込まれてしまうため，部分木の学習は効率的な学習法が必要となる．

3. 提案手法

3.1 確率モデル

我々の提案する木挿入文法の確率モデルは，基本木の確率的生成モデルと，基本木の置換・挿入操作に基づく構文木の確率的生成モデルとで構成される．

3.1.1 基本木の生成モデル

G_X を，根ノードの非終端記号が X である基本木の確率分布とし，G_X の事前分布として Pitman-Yor 過程 [17] を仮定する．このとき，根ノードの非終端記号が X である基本木 e の確率分布は，以下のようにモデル化できる．

$$\begin{aligned} e|X &\sim G_X \\ G_X|d_X,\theta_X &\sim \mathrm{PYP}\left(d_X,\theta_X,P_0\left(\cdot\,|X\right)\right) \end{aligned} \tag{1}$$

ただし，PYP は Pitman-Yor 過程を表す．d_X，θ_X，$P_0\left(\cdot\,|X\right)$ は Pitman-Yor 過程のパラメータであり，それぞれ割引パラメータ（discount parameter），強度パラメータ（strength parameter），基底分布（base distribution）と呼ばれる．Pitman-Yor 過程はベキ則に従う確率分布を生成することができ，言語データの統計的性質と親和性が高いことが知られている [10], [21]．

式 (1) に基づいて基本木 e を i 回生成し，これらを $\mathbf{e}_{1:i} = e_1, e_2, \ldots, e_i$ とする．このとき，$i+1$ 番目の基本木 e_{i+1} の生成確率は，式 (1) の G_X を積分消去することにより以下のように計算される．

$$\begin{aligned} p\left(e_{i+1}|\mathbf{e}_{1:i},X,d_X,\theta_X\right) &= \alpha_{e_{i+1},X} + \beta_X P_0\left(e_{i+1}|X\right) \quad (2)\\ \alpha_{e_{i+1},X} &\equiv \frac{n_{e_{i+1},X} - d_X \cdot t_{e_{i+1},X}}{\theta_X + n_{\cdot,X}} \\ \beta_X &\equiv \frac{\theta_X + d_X \cdot t_{\cdot,X}}{\theta_X + n_{\cdot,X}} \end{aligned}$$

ただし，$n_{e_{i+1},X}$ は，$\mathbf{e}_{1:i}$ のうち e_{i+1} と同じ基本木が生成された回数を表す．$t_{e_{i+1},X}$ は，e_{i+1} のラベルが付与されたテーブルの数を表す．Pitman-Yor 過程のテーブルに関する説明は後述する．また，$n_{\cdot,X} = \sum_e n_{e,X}$，$t_{\cdot,X} = \sum_e t_{e,X}$ である．

式 (2) において，第 1 項目：$\alpha_{e_{i+1},X}$ は基本木 e_{i+1} と同じ種類の基本木がすでに何回生成されたかに基づいて計算される確率，第 2 項目：$\beta_X P_0\left(e_{i+1}\,|X\right)$ は基本木 e_{i+1} のスムージング確率ととらえることができる．スムージング確率は，係数 β_X と基底確率 $P_0\left(e_{i+1}\,|X\right)$ の積で表される．

式 (2) は，中華料理過程（Chinese Restaurant Process）として知られている [21]．中華料理過程では，可算無限個のテーブルを持つ仮想的な中華料理レストランを考え，そこへ客 e が 1 人ずつ入店し，どこかのテーブルへ確率的に着席する過程を考える．1 人目の客 e_1 は必ず 1 番目のテーブルへ着席し，それ以降の客 e_{i+1} $(i \geq 1)$ は，すでに客が着席しているテーブルには確率 $\alpha_{e_{i+1},X}$ で，誰も着席していない新たなテーブルへは確率 β_X で着席する．新たなテーブルへ着席した場合，確率 $P_0\left(e_{i+1}\,|X\right)$ でテーブルにラベルが付与される．つまり，我々のモデルにおいて，テーブルのラベルは何らかの基本木であり，客が着席する動作を基本木の生成と置き換えれば，中華料理過程は式 (2) に従う基本木の生成過程となっている．中華料理過程は，基本木を生成するにつれて確率的にテーブルの数が増加していくモデルである．したがって，あらかじめ基本木の全可能性を保持する必要はなく，データの増加につれて動的に基本木の種類を増加させていけばよい．この性質は，MCMC 法を用いて確率モデルの学習を行う際に，メモリ使用量を抑えることができるという利点がある．

Pitman-Yor 過程の基底分布 P_0 は，基本木 e の基底となる確率分布である．我々は，Cohn らによる木置換文法の確率モデル [5] と同様の基底分布を設定する．具体的には，基本木 e を文脈自由文法の規則（高さが 1 の部分木）に分解し，それらの積として以下のようにモデル化する．

$$\begin{aligned} P_0\left(e\,|X\right) = &\prod_{r\in\mathrm{CFG}(e)} P_{\mathrm{MLE}}\left(r\right) \\ &\times \prod_{A\in\mathrm{LEAF}(e)} s_A \times \prod_{B\in\mathrm{INTER}(e)} \left(1-s_B\right) \end{aligned} \tag{3}$$

ただし，$\mathrm{CFG}(e)$ は，e を高さが 1 となるように分解した部分木の集合で，$P_{\mathrm{MLE}}\left(r\right)$ は構文木コーパスから計算される部分木 r の最尤推定値を表す．また，$\mathrm{LEAF}\left(e\right)$, $\mathrm{INTER}\left(e\right)$ はそれぞれ，e の葉ノードおよび中間ノードにおける非終端記号の集合を表す．s_X は基本木の生成がノード X で停止する確率（停止確率）である．つまり，基底分布 P_0 に基づく基本木 e の生成は，まず根ノード X から始まり，高さが 1 の部分木を結合して部分木を成長させていくという確率モデルである．葉ノードが非終端記号である場合，停止確率 s によって生成を停止するか，または確率 $(1-s)$ でさらに基本木を成長させる．

我々は，初期木と補助木の確率分布を，それぞれ独立に式 (1) で定義する．ただし，基底分布は互いの確率分布で共通である．以降，初期木を e，初期木の確率分布に用いる Pitman-Yor 過程を $\mathrm{PYP}\left(d_X,\theta_X,P_0\left(\cdot\,|X\right)\right)$ とし，補助木を e'，補助木の確率分布に用いる Pitman-Yor 過程を $\mathrm{PYP}\left(d'_X,\theta'_X,P_0\left(\cdot\,|X\right)\right)$ と区別する．

3.1.2 構文木の生成モデル

木挿入文法における構文木の生成は，まず開始記号 S を根ノードとして始まり，すべての葉ノードが終端記号となるまで基本木の置換・挿入操作を繰り返す．本項では，この導出過程の確率モデルを新たに提案する．

構文木の導出過程において，構文木の葉ノードが非終端記号であるときには，必ず初期木による置換操作が起こる．したがって，非終端記号が X である構文木の葉ノードが初期木 e によって置換される確率は，式 (2) より $1\times p\left(e\,|\mathbf{e},X,d_X,\theta_X\right)$ となる．ただし，e の下付き文字 i は

簡単のため省略してある．一方，構文木の内部ノードでは，補助木による挿入操作が確率 a_X で起こると仮定する．つまり，挿入確率は内部ノードの非終端記号 X に依存する．さらに，a_X に対する事前分布としてベータ分布を仮定し，挿入確率 a_X の平滑化を行う．すなわち，$a_X \sim \mathrm{Beta}(b_1, b_2)$ と仮定する．このようなモデル化によって，たとえば動詞句では挿入操作が起こりやすく，名詞句では起こりにくいといった現象をとらえることができる．以上のことから，非終端記号が X である構文木の内部ノードが，根ノードの非終端記号が X である補助木 e' によって置換される確率は $a_X \times p(e'|\mathbf{e}', X, d'_X, \theta'_X)$ とモデル化できる．我々のモデルでは，挿入操作は同じ内部ノードでたかだか 1 度しか起こらないと制約する．

以上をまとめると，基本木の置換・挿入操作に基づく構文木の確率的な生成過程は以下のようになる．

(1) 根ノードの非終端記号を S とする．
(2) 構文木の各ノードについて，
  (a) もし葉ノードかつ非終端記号であれば，初期木 e を確率 $p(e|\mathbf{e}, X, d_X, \theta_X)$ で置換する．
  (b) もし内部ノードであれば，確率 a_X で挿入操作を行う．挿入操作を行う場合，確率 $p(e'|\mathbf{e}', X, d'_X, \theta'_X)$ で補助木を選ぶ．
(3) すべての葉ノードが終端記号であり，かつすべての内部ノードで挿入が起こるかどうかをチェックしたら終わり．そうでなければ (2) へ戻る．

3.2 推定

構文木コーパス $\mathbf{t}$ が与えられたときに，MAP 推定によって確率モデルを学習し，構文木を構成する木挿入文法の基本木を獲得するという問題を考える．このとき，構文木 t を構成する導出過程 $\mathbf{e}$ の事後分布は，ベイズの定理を用いて以下のように計算できる．

$$P(\mathbf{e}|t) \propto P(t|\mathbf{e})\, P(\mathbf{e}) \tag{4}$$

ただし，$P(\mathbf{e})$ は前述の Pitman-Yor 過程に基づく構文木の生成モデルであり，$P(t|\mathbf{e})$ は，導出過程 $\mathbf{e}$ によって得られる構文木が t と一致すれば $P(t|\mathbf{e}) = 1$，そうでなければ $P(t|\mathbf{e}) = 0$ の値をとる．

したがって，構文木 t を生成可能な導出過程のみを考慮して $P(\mathbf{e})$ から基本木のサンプリングを行えば，導出過程 $\mathbf{e}$ の事後分布に従う基本木のサンプルが得られる．$P(\mathbf{e}|t)$ からの効率的なサンプリングを実現するために，まず木挿入文法分解法について説明し，次にブロック化メトロポリス・ヘイスティングスアルゴリズム（Blocked Metropolis-Hastings algorithm，以下ブロック化 MH アルゴリズムと略す）について述べる．

3.2.1 木挿入文法分解

木挿入文法分解は，我々の木挿入文法モデルを MH アルゴリズムにより効率的に学習するための手法である．基本的なアイディアは，木挿入文法の導出過程を等価な文脈自由文法へ変換することである．その結果，文脈自由文法の標準的な学習アルゴリズムである動的計画法をそのまま適用することができる．

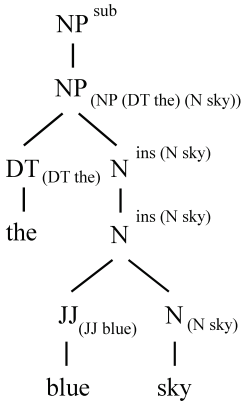

図 4　図 3 の導出過程に木挿入文法分解を適用した例．
Fig. 4　Example derivation of TIG decomposition.

まず初めに，木挿入文法分解の具体例を示す．たとえば，図 3 の導出過程は，まず根ノード NP が初期木 “(NP (DT the) (N sky))” によって置換され，さらに，内部ノード N に補助木 “(N (JJ blue) (N))” が挿入されたものである．これを木挿入文法分解によって文脈自由文法に変換したものが**図 4** である．図 4 において，$\mathrm{NP}^{\mathrm{sub}}$ は NP で置換操作が実行されたことを表し，$\mathrm{NP}_{(\mathrm{NP\ (DT\ the)\ (N\ sky)})}$ は，初期木 “(NP (DT the) (N sky))” を必ず生成する特殊な NP ノードである．したがって，NP ノードが初期木 “(NP (DT the) (N sky))” によって置換されたことを表している．また，$\mathrm{N}^{\mathrm{ins\ (N\ sky)}}$ は，内部ノード $\mathrm{N}_{(\mathrm{N\ sky})}$ で挿入操作が起こったことを表しており，$\mathrm{N}^{\mathrm{ins\ (N\ sky)}}_{(\mathrm{N\ (JJ\ blue)\ N*})}$ は，$\mathrm{N}_{(\mathrm{N\ sky})}$ ノードで補助木 “(N (JJ blue) N*)” が挿入されたことを表す．ここで，図 4 の生成規則を文脈自由文法の規則に分解し，**表 1** のように確率を割り当てると，式 (2) の生成確率に従う導出過程と一致する．表 1 の例では，基本木はすべて式 (2) の第 1 項から生成されたと仮定しているが，第 2 項から生成する場合は $\alpha_{e_i,X}$ や $\alpha'_{e_i,X}$ の代わりに $\beta_{e_i,X}$ や $\beta'_{e_i,X}$ を割り当てればよい．以上のように，木挿入文法の導出過程を図 4 のように変換し，基本木の生成を表 1 のように文脈自由文法の確率の積で表現する方法が木挿入文法分解である．

ここで，木挿入文法分解を用いる動機をより詳細に説明する．Cohn ら [3] が木置換文法のモデルにおいて指摘しているように，事後分布 $P(\mathbf{e}|t)$ に基づく導出過程のサンプルを得ることは，一般には計算コストが非常に高い．これは，基底分布 P_0 が可算無限どおりの基本木に対して 0 より大きな確率を付与するため，基本木の全可能性を明示的に列挙することが非現実的であることに起因する．さらに，木挿入文法は木置換文法と異なり，構文木上の各ノードが

表 1　図 4 の分解規則と確率.
Table 1　Decomposed rules and probabilities of Fig. 4.

分解規則		確率
$\mathrm{NP}^{\mathrm{sub}}$	$\rightarrow \mathrm{NP}_{(\mathrm{NP\ (DT\ the)\ (N\ sky)})}$	$\alpha_{(e=(\mathrm{NP\ (DT\ the)\ (N\ sky)}),\mathrm{NP})}$
$\mathrm{NP}_{(\mathrm{NP\ (DT\ the)\ (N\ sky)})}$	$\rightarrow \mathrm{DT}_{(\mathrm{DT\ the})}\mathrm{N}^{\mathrm{ins\ (N\ sky)}}$	$\left(1-a_{(X=\mathrm{DT})}\right)\times a_{(X=\mathrm{N})}$
$\mathrm{DT}_{(\mathrm{DT\ the})}$	$\rightarrow$ the	1
$\mathrm{N}^{\mathrm{ins\ (N\ sky)}}$	$\rightarrow \mathrm{N}^{\mathrm{ins\ (N\ sky)}}_{(\mathrm{N\ (JJ\ blue)\ N*})}$	$\alpha'_{(e=(\mathrm{N\ (JJ\ blue)\ N*}),\mathrm{N})}$
$\mathrm{N}^{\mathrm{ins\ (N\ sky)}}_{(\mathrm{N\ (JJ\ blue)\ N*})}$	$\rightarrow \mathrm{JJ}_{(\mathrm{JJ\ blue})}\mathrm{N}_{(\mathrm{N\ sky})}$	$\left(1-a_{(X=\mathrm{JJ})}\right)\times 1$
$\mathrm{JJ}_{(\mathrm{JJ\ blue})}$	$\rightarrow$ blue	1
$\mathrm{N}_{(\mathrm{N\ sky})}$	$\rightarrow$ sky	1

基本木の境界ノードか内部ノードなのかに加えて，補助木の末尾ノードかどうかも推定する必要があるために，さらなる計算コストが要求される．しかしながら，我々の基底分布 P_0（式 (3)）は高さが 1 である部分木の生成確率の積である，すなわち，文脈自由文法の規則に分解可能である．さらに，式 (2) の第 1 項目に従う基本木も文脈自由文法の規則に変換可能であり，結果として木挿入文法の導出過程は文脈自由文法で記述可能である．したがって，Johnson ら [11] によって提案された，動的計画法に基づく文脈自由文法の学習法（MH アルゴリズム）をそのまま適用することができる．同様の手法は Cohn ら [3] によって木置換文法のモデルに使用されているが，我々の手法は挿入操作を考慮している点において彼らの手法と異なる．

前述の木挿入文法分解を一般化すると以下のようになる．木挿入文法分解では，式 (2) に従う基本木 e_{i+1} の生成が第 1 項目：$\alpha_{e_{i+1},X}$ からの生成なのか，または第 2 項目：$\beta_X P_0\left(e_{i+1}\,|X\right)$ からの生成なのかを明示的に区別する．第 1 項目からの生成は，以前生成した基本木 $\mathbf{e}_{1:i}$ の中から確率的にいずれかを選択することに相当する．一方で，第 2 項目からの生成は，基底分布 $P_0\left(e_i\,|X\right)$ に従って新たに基本木を生成することに相当する．また，非終端記号 X を以下の 6 種類に分類する．

初期木のノードに使用される非終端記号

- X^{sub}：置換操作が起こるノードであり，非終端記号が X であることを表す．X^{sub} は，確率 $\alpha_{e,X}$ で子ノード $X_{(e)}$ を生成するか，または確率 β_X で子ノード $X_{(\mathrm{base})}$ を生成する．これはつまり，式 (2) の第 1 項目から初期木 e を生成して置換操作を行うのか，または第 2 項目から初期木 e を生成して置換操作を行うのかを表している．
- $X_{(e)}$：式 (2) の第 1 項目から生成された，初期木の根ノードまたは内部ノードの非終端記号を表す．$X_{(e)}$ は，必ず e の情報に基づいて子ノードを生成しなければならない特殊なノードである．たとえば，X を NP とし，e を “NP (DT) (N girl)” とすると，$\mathrm{NP}_{(\mathrm{NP\ (DT\ )\ (N\ girl)})}$ は 2 つの子ノード：$\mathrm{DT}^{\mathrm{sub}}$ と $\mathrm{N}_{(\mathrm{N\ girl})}$ を確率 1 で生成する．同様に，$\mathrm{N}_{(\mathrm{N\ girl})}$ は終端記号 “girl” を確率 1 で生成する．
- $X_{(\mathrm{base})}$：初期木の基底分布から生成された，根ノードまたは内部ノードの非終端記号を表す．$X_{(\mathrm{base})}$ は，初期木の基底分布に従って子ノードを生成する．

補助木のノードに使用される非終端記号

- $X^{\mathrm{ins}(type)}$：挿入操作が起こるノードであり，非終端記号が $X_{(type)}$ であることを表す．ただし，“*type*” は，“base” または e をとる．X^{sub} と同様に，$X^{\mathrm{ins}(type)}$ は確率 $\alpha'_{e,X}$ で子ノード $X^{\mathrm{ins}(type)}_{(e')}$ を生成するか，または確率 β'_X で子ノード $X^{\mathrm{ins}(type)}_{\mathrm{base}}$ を生成する．
- $X^{\mathrm{ins}(type)}_{(e')}$：式 (2) の第 1 項目から生成された，補助木の根ノードまたは内部ノードの非終端記号を表す．$X_{(e)}$ と同様に，補助木 e' の情報に基づいて子ノードを生成する．
- $X^{\mathrm{ins}(type)}_{\mathrm{base}}$：補助木の基底分布から生成された，根ノードまたは内部ノードの非終端記号を表す．補助木の基底分布に従って子ノードを生成する．

以上の分類は，構文木の各ノードで置換操作または挿入操作が起こったかどうか，そのときに式 (2) の第 1 項目から基本木が生成されたか，または第 2 項目から生成されたのかという細かな情報を非終端記号に付与したものと考えられる．このように，各ノードに導出過程に関するすべての情報を付与することで，木挿入文法モデルを表 1 のように文脈自由文法として扱い，動的計画法に基づく効率的な導出過程の推定が可能となる．表 1 の分解規則と確率を一般化したものを付録に示す．

3.2.2　ブロック化 MH アルゴリズム

Johnson らによって提案されたブロック化 MH アルゴリズム [11] を用いると，構文木コーパス $\mathbf{t}$ から我々の確率モデルを学習することができる．MH アルゴリズムは MCMC 法の一種であり，任意の確率分布からランダムサンプルを得る汎用的な手法である．我々の確率モデルでは，導出過程の事後分布 $p\left(\mathbf{e}\,|\mathbf{t},\mathbf{d},\boldsymbol{\theta},\mathbf{s},\mathbf{a}\right)$ に従う基本木のサンプルを得ることが目標となる．これは，前述の木挿入文法分解を利用すれば，構文木 t を文脈自由文法でモデル化し，各ノードの非終端記号 X に $X_{(e)}$，$X_{(\mathrm{base})}$ などの情

報を教師なし学習で付与する問題と等価である．また，モデルのハイパーパラメータ $(\mathbf{d}, \boldsymbol{\theta}, \mathbf{s}, \mathbf{a})$ も同様に MH アルゴリズムで学習することができ，詳細は後述する．

ブロック化 MH アルゴリズムは，構文木 t ごとに以下の 3 ステップで構成される．

(1) 構文木 t の各ノードにおいて内側確率 [12] を計算する．内側確率とは，あるノード X が終端記号列 $\mathbf{w}_{i:j} = w_i, w_{i+1}, \ldots, w_j$ を生成する確率 $p(X \rightarrow \mathbf{w}_{i:j})$ であり，動的計画法により効率的に計算できる．我々のモデルでは，X が X^{sub}，$X_{(e)}$，$X_{(\mathrm{base})}$ などのすべての可能性を考慮して内側確率を計算する．内側確率の計算は，構文木の葉ノードからボトムアップに行われる．

(2) (1) で計算された内側確率に基づいて，図 4 のような導出過程のサンプルを得る．これは，構文木の根ノードからトップダウンに行われる．

(3) MH アルゴリズムによって，サンプルを受理または棄却する．MH アルゴリズムは，マルコフ連鎖が正しい事後分布へ収束するために必要な確率計算である．サンプルが受理された場合は，現在の導出過程を新たなサンプルで置き換える．棄却された場合は，現在の導出過程をそのまま保持する．

上記の 3 ステップを，すべての構文木について順番に繰り返すことで，導出過程の事後分布に基づくサンプルを得ることができる．ブロック化 MH アルゴリズムの詳細は，文献 [3], [11] に示されている．木挿入文法は，文脈自由言語（context-free language）を生成することが知られている [19]．これは，木挿入文法分解によって，我々の木挿入文法モデルが等価な文脈自由文法規則へ変換できることからも明らかである．したがって，上記の MH アルゴリズムに要する計算量のオーダは，文脈自由文法や木置換文法と同じく $\mathcal{O}\left(n^3 \times |G|\right)$ となる．ただし，n は文の単語数，$|G|$ は部分木の数である．

確率モデルのハイパーパラメータ $(\mathbf{d}, \boldsymbol{\theta}, \mathbf{s}, \mathbf{a})$ も同様にサンプリングによって適切な値を推定する．すなわち，各ハイパーパラメータを確率変数と見なし，MCMC 法によりランダムサンプルを得て更新する．そのために，各ハイパーパラメータが以下の確率分布に基づき生成されたとモデル化する．

- $d_X \sim \mathrm{Beta}\,(1,1)$，$d'_X \sim \mathrm{Beta}\,(1,1)$
- $\theta_X \sim \mathrm{Gamma}\,(0.1, 10.0)$，$\theta'_X \sim \mathrm{Gamma}\,(0.1, 10.0)$
- $s_X \sim \mathrm{Beta}\,(1,1)$
- $a_X \sim \mathrm{Beta}\,(b_1, b_2)$

ただし，(b_1, b_2) の値は，実験の際に様々な値を設定して，挿入操作の効果を検証することとする．また，d_X と θ_X の値は，補助変数によるサンプリング法 [20] を用いて更新する．

4. 実験

4.1 設定

英語の構文木コーパスである WSJ Penn Treebank [13] および British National Corpus (BNC) Treebank [*1] を用いて提案手法の評価を行った．Penn Treebank コーパスは，Penn-A（学習用データ：セクション 2，テスト用データ：セクション 22）と Penn-B（学習用データ：セクション 2 から 21 まで，テスト用データ：セクション 23）の 2 種類を用いた．各セクションは約 2,000 文の構文木で構成されており，Penn-A は約 2,000 文の学習用データ，Penn-B は約 40,000 文の学習用データである．これは，Cohn ら [5] による木置換文法の実験設定と同じであり，Penn-A は言語資源の少ない場合，Penn-B は言語資源の多い場合を想定している．また，BNC コーパスは 1,000 文の構文木データであり，最初の 900 文を学習用データ，残りの 100 文をテスト用データとした．

データの前処理として，まずコーパス中の全構文木を “CENTER-HEAD” 法 [14] により二分木へ変換した．また，コーパス中に 1 度しか現れない単語は，単純なルールによって “UNKNOWN”，“DIGIT”，“PROPER_NOUN” のいずれかに置き換えた．実験では，それぞれの学習データに対して MH アルゴリズムを用い，確率モデルの学習を行った．また，MH アルゴリズムで獲得された木挿入文法の基本木：X を用いて構文解析器を作成し，構文木の情報を取り除いたテストデータに構文解析を行って精度を評価した．構文解析の精度評価は，EVALB [*2] で計算されるブラケティング（bracketing）F 値 [6] を用いた．ただし，MH アルゴリズムは確率的に最適解を探索する手法であるため，MH アルゴリズムを独立に 10 回試行し，ブラケティング F 値の平均値を構文解析の精度とした．本実験で用いた計算機は，CPU が Xeon 5600 3.33 GHz，メモリが 48 GB である．

4.2 結果

4.2.1 Penn-A データの実験

Penn-A データを用いて提案モデルの評価を行った．まず，MH アルゴリズムの反復回数と対数尤度，構文解析精度の関係を**表 2** に示す．反復回数を増やしていくと提案モデルの対数尤度は増加したが，構文解析精度は反復回数が 1,000 回より多ければほぼ同等の結果であった．また，1 反復にかかる計算時間は約 2.5 秒で，反復回数が増えてもほぼ変化がない．したがって，Penn-A データと同じような規模のデータでは，現実時間内に提案モデルの学習を完了することができる．

[*1] http://nclt.computing.dcu.ie/~jfoster/resources/
[*2] http://nlp.cs.nyu.edu/evalb/

表 3　Penn-A データでの構文解析結果（MH アルゴリズムの反復 1,000 回）．“# 基本木” は学習後に得られた基本木が何種類であるかを表し，“# 補助木” は得られた基本木の中で補助木が何種類であるかを表す

Table 3　Parsing results on Penn-A data (the number of MH iterations: 1,000).

文法モデル	(b_1, b_2)	# 基本木（# 補助木）	F 値
文脈自由文法	-	5,957 (0)	65.0
木置換文法（$a_X = 0$ for all X）	-	9,135 (0)	77.9
木挿入文法（提案モデル）	(1,1)	7,928 (3)	**79.3**
	(100,1)	7,462 (16)	75.4
	(100,100)	8,057 (15)	78.0
Berkley Parser [16]		-	77.9
木置換文法（Cohn ら [3] の結果）		-	78.4

表 2　MH アルゴリズムの反復回数と対数尤度，構文解析精度の関係（Penn-A データ）

Table 2　Experimental results of log-likelihood and parsing accuracy on Penn-A data.

反復回数	対数尤度	時間（sec）	F 値
500	−282,418.4	1,258	78.2
1,000	−276,171.9	2,583	79.3
3,000	−273,591.3	7,681	79.1
5,000	−272,766.8	12,794	79.0

次に，Penn-A データを用いて提案モデルの詳細な評価を行った．**表 3** に結果を示す．表中の木置換文法の結果は，提案モデルの挿入確率 a_X をすべての非終端記号 X において 0 に設定することにより得た．また，(b_1, b_2) は提案モデルにおける挿入操作の起こりやすさを調節するベータ分布のパラメータである．b_1 に大きな値を設定すると補助木による挿入操作が起こる確率を高め，b_2 に大きな値を設定すると挿入操作を抑制する働きがある．実際，$b_1 = 100$ のときに補助木設定した場合は，$b_1 = 1$ と比較して獲得された補助木の数が増加していることが分かる．提案モデルでは，獲得される基本木の数や構文解析精度が (b_1, b_2) の値に影響されやすいことが分かる．

木置換文法と木挿入文法のモデルは，文脈自由文法のモデルと比較して構文解析の精度がきわめて高いことが分かる．文脈自由文法は高さが 1 の部分木のみを使用して構文木を生成するのに対して，木置換文法と木挿入文法では任意の大きさの部分木を使用できる．したがって，これらのモデルでは頻出する構文パターンを 1 つの部分木として学習し，高い確率を付与できる．その結果，木置換文法や木挿入文法は文脈自由文法よりも基本木の数は増加するが，より高精度を実現できていると考えられる．

木置換文法と木挿入文法を比較すると，木挿入文法は適切な (b_1, b_2) の値を設定すれば，木置換文法よりも約 1.4 ポイントの構文解析精度向上が確認できた．また，基本木の数を比較すると，木挿入文法は木置換文法よりも約 18%少なかった．したがって，補助木による挿入操作が汎用性の高い構文パターンの獲得に寄与していると考えられる．実際に獲得された補助木の例は後の実験結果で示す．

最後に，提案モデルと既存モデルとを比較した．提案モデルは，現在標準的に使用されている Berkley Parser [16] や，Cohn らの木置換文法モデル [3] よりも高精度であることが分かる．したがって，本手法により，Penn-A データと同等の規模のデータでは非常に高精度な構文解析器が得られることが期待できる．

4.2.2　Penn-B，BNC データの実験

次に，BNC，Penn-B データを用いて提案モデルの構文解析結果を評価した．**表 4** に結果を示す．BNC データでは，提案モデルが最も良い構文解析精度であった．これは，Penn-A データと同様である．Penn-B データでは，Penn-A や BNC データと同様に，木置換文法と木挿入文法のモデルは文脈自由文法のモデルよりもきわめて高精度であり，木挿入文法はわずかに木置換文法の結果を上回った．これは，学習用データが増加したことにより，木置換文法でも多様な構文パターンを獲得できるようになったためだと考えられる．しかしながら，基本木の数は木挿入文法のほうが約 20%少なく，コンパクトな文法獲得のためには挿入操作が必要であることが分かる．

他手法と比較すると，提案モデルは Post らの木置換文法モデル [18] による結果を大きく上回り，Cohn らの木置換文法モデル [3] と同等の結果であった．Penn-A データでの構文解析結果と比較すると，Penn-B データでは学習データの量が約 10 倍に増加することにより，学習データから獲得される部分木はテストデータに現れる部分木のパターンをほぼ網羅できていると考えられる．したがって，挿入操作を用いなければ生成できない構文パターンは減少し，置換操作のみでもテストデータの構文木を生成できるようになる．その結果，木置換文法と木挿入文法の精度はほぼ同等であったと考えられる．ただし，木挿入文法では付加要素を補助木として区別することができるため，文章要約などのアプリケーションで文の付加要素のみを削除したい場合，木挿入文法は木置換文法よりも有用性が高いと考えられる．

表 4　BNC（MH アルゴリズムの反復 1,000 回），Penn-B データ（MH アルゴリズムの反復 3,500 回）での構文解析結果．*単語数が 40 以下の文のみの結果

Table 4　Parsing results on BNC and Penn-B data (the number of MH iterations: 1,000).

学習データ	文法モデル	# 基本木（# 補助木）	F 値
BNC	文脈自由文法	3,865 (0)	54.0
	木置換文法（$a_X = 0$ for all X）	5,203 (0)	67.7
	木挿入文法（提案モデル）	4,976 (7)	**69.1**
Penn-B	文脈自由文法	35,374 (0)	71.0
	木置換文法（$a_X = 0$ for all X）	80,026 (0)	85.0
	木挿入文法（提案モデル）	64,372 (12)	**85.3**
	木置換文法（Post ら [18] の結果）	-	82.6*
	木置換文法 [3]（Cohn ら [18] の結果）	-	**85.3**

表 5　Penn-B データから得られた補助木の例と頻度．非終端記号の上線は，構文木の二分化によって生成されたノードであることを表す．また，“NP” は名詞句，“ADVP” は副詞句，“RB” は副詞，“PP” は前置詞句，“FRAG” はフラグメント，“VP” は動詞句，“QP” は数量詞，“S” は文全体を表す非終端記号である

Table 5　Examples of lexicalized auxiliary trees obtained from our model in Penn-B data experiments. Nonterminal symbols created by binarization are shown with an over-bar.

補助木	頻度
(NP (NP) (ADVP (RB aloft)))	3
(NP (N$\bar{\text{P}}$) (ADVP (RB respectively)))	16
(PP (P$\bar{\text{P}}$) (, ,))	67
(FRAG (, ,) (FR$\bar{\text{A}}$G))	21
(VP (V$\bar{\text{P}}$) (RB then))	9
(VP (V$\bar{\text{P}}$) (RB not))	23
(QP (Q$\bar{\text{P}}$) (IN of))	6
(S ($\bar{\text{S}}$) (: ;))	32
(S ($\bar{\text{S}}$) (RB so))	4

また，Post らのモデルは Dirichlet 過程に基づいて部分木の確率モデルを定義しているが，構文木を二分木へ変換しないという点で我々の手法や Cohn らの手法と異なる．多分木の構文木を木置換文法でモデル化すると，修飾句などの任意要素が含まれる句をそのまま規則として学習するため，汎用性の高い規則を獲得することができない．したがって，Post らの手法は Penn-B データのような規模の大きな学習データでも精度があまり高くないと考えられる．

次に，Penn-B データから得られた補助木の例とその頻度を**表 5** に示す．主に，記号（“, ”，“;”）や副詞（RB）となる単語が挿入操作によって構文木へ挿入されやすいことが分かる．本結果は，英語において記号や副詞が文の様々な位置へ挿入されやすいという知見と合致し，統計的な観点からも，これらを補助木の挿入ととらえることで全体の基本木の種類を少なくすることに寄与している．

また，獲得された補助木の種類は，初期木と比較して非常に少ないことが分かる．以下に理由を考察する．木挿入文法では，名詞や動詞などの文の骨格となる要素は置換操作によって，また，修飾要素や記号などの付加要素は挿入操作によって文の構造を導出することが期待できる．しかし，統計モデルの観点からは，たとえ修飾要素や記号であったとしても，それらが定形パターンとして学習データに頻出する場合には置換操作として学習し，稀に発生する付加要素は挿入操作として学習することによってモデル全体の尤度が最大になる．本実験で用いた学習データはすべて新聞記事の文であり，副詞や形容詞の挿入パターンに関する多様性はあまり高くないため，多くの副詞や形容詞は補助木ではなく初期木として獲得されたと考えられる．

また，Cohn ら [3] が指摘しているように，MCMC 法を用いて Pitman-Yor 過程または Dirichlet 過程に基づく構文木の確率モデルを学習する場合，学習データの規模が大きいと局所解にとどまってしまうという問題がある．今後，今回の学習に用いた MH 法に，スライスサンプリング（slice sampling）[15] などの効率的なサンプリング法を組み合わせることで局所解をうまく回避することができれば，現在よりも高精度な構文解析器を実現できる可能性がある．また，英語以外の言語や Web などの CGM データでは様々な要素の挿入が頻出する場合があり，木挿入文法によって汎用性の高い構文パターンを学習できる可能性がある．

5.　おわりに

我々は，Pitman-Yor 過程に基づく木挿入文法の確率モデルを構築し，構文木コーパスから統計的に基本木を獲得する学習法を提案した．提案モデルは，膨大な可能性の部分木の確率分布を表現するために Pitman-Yor 過程を利用し，挿入操作を含む確率モデルの効率的な学習のために木挿入文法分解法を新たに提案した．提案手法を英語の構文木コーパスに適用したところ，比較的少量の学習用データでは文脈自由文法や木置換文法と比較して構文解析精度の向上を実現した．また，学習用データの規模が大きいときは，提案モデルは木置換文法のモデルと比較して約 20%コンパクトでありながら，ほぼ同等の構文解析結果を達成

した．

参考文献

[1] Chen, J., Bangalore, S. and Vijay-Shanker, K.: Automated extraction of Tree-Adjoining Grammars from treebanks, *Natural Language Engineering*, Vol.12, No.03, pp.251–299 (2006).

[2] Chiang, D.: *Statistical Parsing with an Automatically Extracted Tree Adjoining Grammar*, chapter 16, pp.299–316, CSLI Publications (2003).

[3] Cohn, T. and Blunsom, P.: Blocked Inference in Bayesian Tree Substitution Grammars, *Proc. Association for Computational Linguistics* (*ACL*) *2010 Conference Short Papers*, Uppsala, Sweden, pp.225–230, Association for Computational Linguistics (2010).

[4] Cohn, T., Blunsom, P. and Goldwater, S.: Inducing Tree-Substitution Grammars, *Journal of Machine Learning Research* (2011).

[5] Cohn, T., Goldwater, S. and Blunsom, P.: Inducing Compact but Accurate Tree-Substitution Grammars, *Proc. Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (*HLT-NAACL*), Boulder, Colorado, pp.548–556, Association for Computational Linguistics (2009).

[6] Collins, M.: Head-Driven Statistical Models for Natural Language Parsing, *Computational Linguistics*, Vol.29, No.4, pp.589–637 (2003).

[7] DeNeefe, S. and Knight, K.: Synchronous Tree Adjoining Machine Translation, *Proc. 2009 Conference on Empirical Methods in Natural Language Processing* (*EMNLP*), Singapore, pp.727–736, Association for Computational Linguistics (2009).

[8] DeNeefe, S., Knight, K. and Vogler, H.: A decoder for probabilistic synchronous tree insertion grammars, *Proc. 2010 Workshop on Applications of Tree Automata in Natural Language Processing*, pp.10–18, Association for Computational Linguistics (2010).

[9] Johnson, M., Griffiths, T. and Goldwater, S.: Adaptor grammars: A framework for specifying compositional nonparametric Bayesian models, *Advances in Neural Information Processing Systems* (*NIPS*), Vol.19, p.641 (2007).

[10] Johnson, M. and Goldwater, S.: Improving nonparameteric Bayesian inference: experiments on unsupervised word segmentation with adaptor grammars, *Proc. Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (*HLT-NAACL*), Boulder, Colorado, pp.317–325, Association for Computational Linguistics (2009).

[11] Johnson, M., Griffiths, T. and Goldwater, S.: Bayesian Inference for PCFGs via Markov Chain Monte Carlo, *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference* (*HLT-NAACL*), Rochester, New York, pp.139–146, Association for Computational Linguistics (2007).

[12] Lari, K. and Young, S.: Applications of stochastic context-free grammars using the inside-outside algorithm, *Computer Speech & Language*, Vol.5, No.3, pp.237–257 (1991).

[13] Marcus, M.P., Santorini, B. and Marcinkiewicz, M.A.: Building a Large Annotated Corpus of English: The Penn Treebank, *Computational Linguistics*, Vol.19, No.2, pp.313–330 (1993).

[14] Matsuzaki, T., Miyao, Y. and Tsujii, J.: Probabilistic CFG with latent annotations, *Proc. 43rd Annual Meeting on Association for Computational Linguistics* (*ACL*), pp.75–82, Association for Computational Linguistics (2005).

[15] Neal, R.M.: Slice sampling, *Annals of Statistics*, pp.705–741 (2003).

[16] Petrov, S., Barrett, L., Thibaux, R. and Klein, D.: Learning Accurate, Compact, and Interpretable Tree Annotation, *Proc. 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics* (*ICCL-ACL*), Sydney, Australia, pp.433–440, Association for Computational Linguistics (2006).

[17] Pitman, J. and Yor, M.: The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator, *The Annals of Probability*, Vol.25, No.2, pp.855–900 (1997).

[18] Post, M. and Gildea, D.: Bayesian Learning of a Tree Substitution Grammar, *Proc. ACL-IJCNLP 2009 Conference Short Papers*, Suntec, Singapore, pp.45–48, Association for Computational Linguistics (2009).

[19] Schabes, Y. and Waters, R.: Tree insertion grammar: a cubic-time, parsable formalism that lexicalizes context-free grammar without changing the trees produced, *Fuzzy Sets and Systems*, Vol.76, No.3, pp.309–317 (1995).

[20] Teh, Y.W.: A Bayesian Interpretation of Interpolated Kneser-Ney, Technical Report TRA2/06, School of Computing, National University of Singapore (2006).

[21] Teh, Y.W.: A Hierarchical Bayesian Language Model based on Pitman-Yor Processes, *Proc. 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics* (*ICCL-ACL*), pp.985–992 (2006).

[22] Xia, F.: Extracting tree adjoining grammars from bracketed corpora, *Proc. 5th Natural Language Processing Pacific Rim Symposium* (*NLPRS*), pp.398–403 (1999).

付　　録

A.1　木挿入文法分解

3.2.1 節で述べた木挿入文法分解の詳細について，**表 A·1** に示す．

表 A·1　木挿入文法分解の規則と確率．(a) X^{sub} と $X^{\mathrm{ins}(type)}$ の場合．X^{sub} と $X^{\mathrm{ins}(type)}$ は必ず同じ非終端記号の子ノードを 1 つ生成する．(b) $X_{(e)}$ の場合．childL (e) と childR (e) はそれぞれ，e の左と右における子ノード以下の部分木を表す．たとえば $e =$ (NP (DT the) (N girl)) のとき，childL $(e) =$ (DT the)，childR $(e) =$ (N girl) である．また，w は終端記号を表す．(c) $X_{(\mathrm{base})}$ の場合．なお，$X^{\mathrm{ins}(type)}_{(\mathrm{base})}$ と $X^{\mathrm{ins}(type)}_{(e)}$ の場合も同様に，木挿入文法分解の規則と確率を定義できる．$^*\hat{P}(X \to w)$ は学習データから計算される部分木 $X \to w$ の最尤推定値を表す．$\hat{P}(X \to Y)$, $\hat{P}(X \to YZ)$ も同様である．

Table A·1　Decomposed rules and probabilities. (a) X^{sub} and $X^{\mathrm{ins}(type)}$. X^{sub} and $X^{\mathrm{ins}(type)}$ generate unary nonterminal child nodes. (b) $X_{(e)}$. childL (e) and childR (e) are left and right child of e, respectively. (c) $X_{(\mathrm{base})}$. $^*\hat{P}(X \to w)$ is a maximum likelihood estimate of w given X.

分解規則		確率
X^{sub}	$\to X_{(\mathrm{base})}$	β_X
X^{sub}	$\to X_{(e)}$	$\alpha_{e,X}$
$X^{\mathrm{ins}(type)}$	$\to X^{\mathrm{ins}(type)}_{(\mathrm{base})}$	β'_X
$X^{\mathrm{ins}(type)}$	$\to X^{\mathrm{ins}(type)}_{(e)}$	$\alpha'_{e,X}$

(a)

分解規則		確率		
$X_{(e)} \to w$		1	(if $w =$ child (e))	
$X_{(e)} \to Y_{(\mathrm{child}(e))}$		$(1 - a_Y)$		
$X_{(e)} \to Y^{\mathrm{adj}(\mathrm{child}(e))}$		a_Y		
$X_{(e)} \to Y_{(\mathrm{childL}(e))}$	$Z_{(\mathrm{childR}(e))}$	$(1 - a_Y)$	$\times$	$(1 - a_Z)$
$X_{(e)} \to Y_{(\mathrm{childL}(e))}$	$Z^{\mathrm{ins}(\mathrm{childR}(e))}$	$(1 - a_Y)$	$\times$	a_Z
$X_{(e)} \to Y^{\mathrm{ins}(\mathrm{childL}(e))}$	$Z_{(\mathrm{childR}(e))}$	a_Y	$\times$	$(1 - a_Z)$
$X_{(e)} \to Y^{\mathrm{ins}(\mathrm{childL}(e))}$	$Z^{\mathrm{ins}(\mathrm{childR}(e))}$	a_Y	$\times$	a_Z

(b)

分解規則		確率			
$X_{(\mathrm{base})} \to w$		$\hat{P}(X \to w)^*$			
$X_{(\mathrm{base})} \to Y^{\mathrm{sub}}$		$\hat{P}(X \to Y) \times$	s_Y		
$X_{(\mathrm{base})} \to Y_{(\mathrm{base})}$		$\hat{P}(X \to Y) \times$	$(1 - s_Y)(1 - a_Y)$		
$X_{(\mathrm{base})} \to Y^{\mathrm{ins}(type)}$		$\hat{P}(X \to Y) \times$	$(1 - s_Y)\, a_Y$		
$X_{(\mathrm{base})} \to Y^{\mathrm{sub}}$	Z^{sub}	$\hat{P}(X \to YZ) \times$	s_Y	$\times$	s_Z
$X_{(\mathrm{base})} \to Y^{\mathrm{sub}}$	$Z_{(\mathrm{base})}$	$\hat{P}(X \to YZ) \times$	s_Y	$\times$	$(1 - s_Z)(1 - a_Z)$
$X_{(\mathrm{base})} \to Y^{\mathrm{sub}}$	$Z^{\mathrm{ins}(type)}$	$\hat{P}(X \to YZ) \times$	s_Y	$\times$	$(1 - s_Z)\, a_Z$
$X_{(\mathrm{base})} \to Y_{(\mathrm{base})}$	Z^{sub}	$\hat{P}(X \to YZ) \times$	$(1 - s_Y)(1 - a_Y)$	$\times$	s_Z
$X_{(\mathrm{base})} \to Y_{(\mathrm{base})}$	$Z_{(\mathrm{base})}$	$\hat{P}(X \to YZ) \times$	$(1 - s_Y)(1 - a_Y)$	$\times$	$(1 - s_Z)(1 - a_Z)$
$X_{(\mathrm{base})} \to Y_{(\mathrm{base})}$	$Z^{\mathrm{ins}(type)}$	$\hat{P}(X \to YZ) \times$	$(1 - s_Y)(1 - a_Y)$	$\times$	$(1 - s_Z)\, a_Z$
$X_{(\mathrm{base})} \to Y^{\mathrm{ins}(type)}$	Z^{sub}	$\hat{P}(X \to YZ) \times$	$(1 - s_Y)\, a_Y$	$\times$	s_Z
$X_{(\mathrm{base})} \to Y^{\mathrm{ins}(type)}$	$Z_{(\mathrm{base})}$	$\hat{P}(X \to YZ) \times$	$(1 - s_Y)\, a_Y$	$\times$	$(1 - s_Z)(1 - a_Z)$
$X_{(\mathrm{base})} \to Y^{\mathrm{ins}(type)}$	$Z^{\mathrm{ins}(type)}$	$\hat{P}(X \to YZ) \times$	$(1 - s_Y)\, a_Y$	$\times$	$(1 - s_Z)\, a_Z$

(c)

進藤 裕之　（正会員）

2009年早稲田大学大学院先進理工学研究科修士課程修了．同年 NTT 入社．統計的自然言語処理の研究に従事．現在，NTT コミュニケーション科学基礎研究所研究員．ACL Best Paper Award 受賞（2012 年）．ACL 会員．

藤野 昭典　（正会員）

1995 年京都大学工学部精密工学科卒業．1997 年同大学大学院修士課程修了．2009 年同大学院博士課程修了．博士（情報学）．1997 年 NTT 入社．機械学習，テキスト処理等の研究に従事．現在，NTT コミュニケーション科学基礎研究所主任研究員．電子情報通信学会 PRMU 研究奨励賞（2004 年度），FIT 論文賞（2005 年）等受賞．電子情報通信学会，IEEE 各会員．

永田 昌明　（正会員）

1987 年京都大学大学院工学研究科修士課程修了．工学博士．同年 NTT 入社．1989 年から 4 年間 ATR 自動翻訳電話研究所へ出向．1999 年から 1 年間 AT&T 研究所客員研究員．統計的自然言語処理の研究に従事．現在，NTT コミュニケーション科学基礎研究所主幹研究員．情報処理学会奨励賞（1991 年），情報処理学会論文賞（1995 年），人工知能学会研究奨励賞（1995 年）等受賞．電子情報通信学会，人工知能学会，言語処理学会，ACL 各会員．