情報処理学会論文誌　数理モデル化と応用　Vol.7　No.1　34–43 (Mar. 2014)

統計的文法獲得モデルのための擬似部分木ブロック化サンプリング法

進藤 裕之[1,a)]　松本 裕治[2]　永田 昌明[1]

受付日 2013年4月27日，再受付日 2013年6月7日 / 2013年8月1日，
採録日 2013年8月22日

概要：自然言語処理分野における統計的文法獲得では，確率文法モデルの学習に Gibbs サンプリング法が広く用いられている．しかしながら，木構造データを扱う場合には，Gibbs サンプリング法のように変数の値を 1 つずつ順番に更新していく方法では局所解にとどまりやすく，十分に尤度の高い解を得られないという問題がある．この問題を解決するために，我々は新たな部分木の擬似ブロック化サンプリング法を提案する．本手法は，データ中に現れる共通の部分木をまとめてブロック化し，ブロックに含まれる変数の同時分布からサンプリングを行う．さらに，その部分木ブロック化サンプラを従来のマルコフ連鎖モンテカルロ法と組み合わせて交互に実行することによって，より尤度の高い文法規則を効率良く探索できる．ただし，本手法は，文法規則の事後分布を均衡分布とする正確なマルコフ連鎖からのサンプラを構成するものではなく，その近似分布に基づく推論となるため，擬似ブロック化サンプリング法と称する．シンボル細分化文脈自由文法を用いて統計的文法獲得の実験を行ったところ，提案手法は既存手法よりも尤度の高い文法規則が獲得できることを確認した．

キーワード：統計的文法獲得，マルコフ連鎖モンテカルロ法，ブロック化サンプリング

Pseudo Blocked Subtree Sampler for Statistical Grammar Induction

Hiroyuki Shindo[1,a)]　Yuji Matsumoto[2]　Masaaki Nagata[1]

Received: April 27, 2013, Revised: June 7, 2013/August 1, 2013,
Accepted: August 22, 2013

Abstract: Gibbs sampler is widely used for statistical grammar induction in natural language processing. However, by sampling only one variable at a time, the sampler suffers from local optimum due to the strong dependency between variables of tree structure. In this paper, we propose a pseudo blocked subtree sampler to tackle this problem. Our sampler collects the same type of subtrees for each iteration and updates them simultaneously. Further, our method iterates the blocked subtree sampler and conventional Markov chain Monte Carlo (MCMC) sampler alternately to search better grammatical rules efficiently. We refer to our method as a pseudo blocked sampler since it generates grammatical rules from an approximation to the given distribution. The experimental results of grammar induction show that our method achieves better performance compared with conventional methods.

Keywords: statistical grammar induction, Markov chain Monte Carlo methods, blocked sampling

1. はじめに

自然言語処理分野における文法獲得とは，日本語や英語などの文または構文木のデータから，コンピュータを用いて自動的に文法規則を獲得することである．たとえば，文法モデルとして文脈自由文法を用いた場合，文法規則は S → VP NP のような深さ 1 の木構造として定義される．獲

[1] NTT コミュニケーション科学基礎研究所
NTT Communication Science Laboratories, Kyoto 619–0237, Japan
[2] 奈良先端科学技術大学院大学
Nara Institute of Science and Technology, Ikoma, Nara 630–0192, Japan
[a)] shindo.hiroyuki@lab.ntt.co.jp

得された文法規則は，構文解析器や言語モデルとして，機械翻訳や自動要約システムなどに応用されている [5], [6]．従来より，Penn Treebank [11] などの構文木コーパスから，確率文法モデルを用いて統計的に文法規則を獲得する方法が提案されてきた [3], [14], [16]．統計的手法による文法獲得は，人手で作成されたルールによる発見的手法と比較して，言語や構文木のアノテーション仕様に大きく依存しないため様々なデータに適用できるという利点がある．

確率文法モデルの学習法として，Gibbs サンプリング法 [7] が広く用いられている．Gibbs サンプリング法はマルコフ連鎖モンテカルロ（MCMC）法の一種であり，与えられた確率分布を均衡分布とするマルコフ連鎖を利用して，その確率分布に従うサンプルを生成することができる．統計的文法獲得では，構文木が与えられたもとでの文法規則の事後確率分布に従うサンプルを生成し，事後確率が最大となる文法規則を探索する [4], [15], [16], [17]．Gibbs サンプリング法の特徴は，複数の確率変数の同時確率分布から直接サンプルを生成するのではなく，変数を 1 つずつ順番に巡回してサンプリングを行うという点にある．そのため，文法獲得で用いられる多くの確率文法モデルに対して単純な学習アルゴリズムを与え，汎用性が高いという利点がある．一方，確率文法モデルでは，木構造データに起因する変数間の強い相互依存性のため，変数の値を 1 つずつサンプリングする方法では局所解にとどまりやすく，十分に尤度の高い解を得られないという問題点が指摘されている [3]．この問題に対する一般的な改善策として，複数の変数をまとめて同時にサンプリングを行うブロック化 MCMC 法が提案されている [2], [9]．しかしながら，これらの方法は，特定の文法理論や確率モデルに特化したアルゴリズムであったり，確率変数が二値であることを想定していたりするなど，使用するうえで様々な制限があった．

上記の問題点を解決するために，本稿では統計的文法獲得のための新たな擬似ブロック化サンプリング法を提案する．我々の狙いは，Gibbs サンプリング法のような汎用性を失わずに，なるべく多くの変数を同時にサンプリングする方法を構築することである．提案手法は，既存のブロック化 MCMC 法と異なり，Gibbs サンプリング法などの一般的な MCMC 法と，ブロック化サンプリングを行うアルゴリズムとを独立に構築して，それらを交互に実行するというアプローチをとる．ただし，本手法で用いるブロック化サンプリングは，文法規則の事後分布を均衡分布とする正確なマルコフ連鎖からのサンプラを構成するものではなく，その近似分布に基づく推論となるため，擬似ブロック化サンプリング法と称する．また，提案手法では，従来手法のように，どの変数をまとめてサンプリングするのかという定義を事前に与えるのではなく，適切なブロックを自動的に発見して同時にサンプリングを行う．したがって，特定の文法モデルに依存せず，多値変数も扱うことができる．

具体的な文法モデルとして，近年の高精度な構文解析器に使われているシンボル細分化文脈自由文法モデルに対して提案手法を適用したところ，提案手法は，データ量にかかわらず，Gibbs サンプリング法や従来のブロック化サンプリング法よりも尤度の高い文法規則が獲得できることを確認した．

全体の構成は以下のとおりである．2 章では，統計的文法獲得および本稿で扱う具体的な確率文法モデルについて概説する．3 章では，我々の提案する擬似部分木ブロック化サンプリング法について説明する．また，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と考察を述べる．5 章では，提案手法の関連研究について述べ，最後に 6 章で結論を述べる．

2. 統計的手法による文法獲得

本章では，提案手法の前提となる，Gibbs サンプリング法による統計的文法獲得について概説する．本研究では，構文木コーパスはあらかじめ与えられるものとして，構文木コーパスから文法モデルの基本木（elementary trees）を統計的に推定するという問題を考える．基本木とは，文法規則の基本単位となる部分木のことを指す．たとえば，文脈自由文法では，NP → NP DT のような深さ 1 の部分木が基本木である．**図 1** に構文木の例を，**図 2** に，図 1 の構文木における文脈自由文法の基本木を示す．単純な文脈自由文法では，構文木を深さ 1 の部分木に分割すれば基本木の集合が一意に定まるが，それ以外の文法モデルでは，異なる基本木の組合せが同一の構文木を構成する可能性が

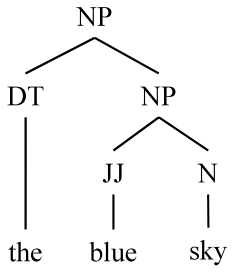


図 1　構文木の例
Fig. 1　Example parse tree.

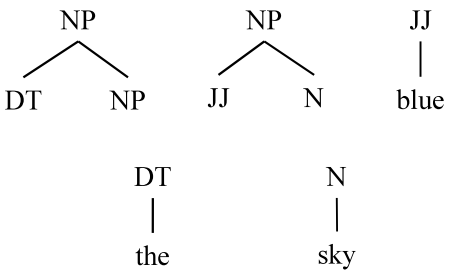


図 2　文脈自由文法の基本木の例
Fig. 2　Example elementary tree of context-free grammars.

ある．そのため，確率文法モデルを用いてもっともらしい基本木の組合せを統計的に推定する必要がある．

2.1　確率文法モデル

構文木 t が与えられたもとでの基本木の集合 $\mathbf{e} = \{e_1, e_2, \ldots\}$ の事後確率は，ベイズの定理によって以下のように計算される．

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

ただし，$P(t|\mathbf{e})$ は，基本木 $\mathbf{e}$ が構成する全体の木構造が構文木 t と一致したときに 1，そうでないときは 0 の値をとる確率分布である．また，$P(\mathbf{e})$ は基本木の確率生成モデルである．

提案手法は，任意の基本木の確率モデル $P(\mathbf{e})$ の学習に適用することが可能であるが，本稿では具体例として，近年の高精度な構文解析器に用いられている確率シンボル細分化文脈自由文法を取り上げる [12], [14]．

シンボル細分化文脈自由文法は，構文木の各ノードに付与されたシンボル（非終端記号）が細分化された文脈自由文法である．シンボルを細分化することにより，たとえば同じ NP（名詞句）のタグが付与されていたノードを，NP-0（文の主語になりやすい名詞句）と，NP-1（文の目的語になりやすい名詞句）のように精緻化できる．**図 3** (a) に，図 1 の構文木に対するシンボル細分化文脈自由文法の導出過程の例を示す．シンボル細分化文脈自由文法では，基本木 e は $A_x \to B_y C_z$ の形式をとる．ただし，A，B，C は NP や VP などのシンボルを表し，x，y，z は細分化カテゴリ（$0, 1, \ldots$）である．シンボル細分化文脈自由文法の確率モデルは，ノンパラメトリックベイズモデルの一種である Pitman-Yor 過程を用いて，以下のように定式化できる [10]．

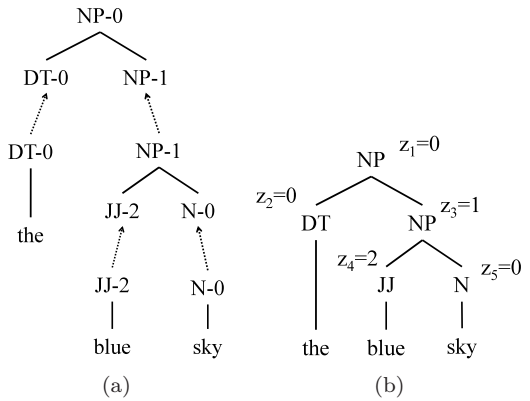


図 3　(a) シンボル細分化文脈自由文法の導出過程の例．点線の矢印は，基本木が結合する過程を表す．(b) (a) の導出過程を，構文木のノードに割り当てた変数で表現したもの

Fig. 3　(a) Example derivation of symbol-refined context-free grammars. The dotted line represents a substitution operation. (b) Latent variable representation of (a).

$$e|A_x \sim G_{A_x}$$
$$G_{A_x} \sim \mathrm{PYP}\left(d_{A_x}, \theta_{A_x}, P_0(\cdot|A_x)\right)$$

ただし，A_x は基本木 e の根ノードのシンボルである．PYP は Pitman-Yor 過程を表し，d_{A_x}，θ_{A_x} は Pitman-Yor 過程のパラメータを表す．P_0 は基底確率分布で，基本木のバックオフ確率を与える．

本研究では，基底確率を以下のように定義する．

$$P_0(e|A_x) = P_{\mathrm{MLE}}(A \to BC) \times \frac{1}{|B|} \times \frac{1}{|C|}$$

ただし，$A \to BC$ は，A_x，B_y，C_z の各シンボルの細分化情報を取り除いた部分木であり，P_{MLE} は構文木コーパスから計算される最尤推定量を表す．また，$|B|$，$|C|$ はそれぞれ，B と C の細分化カテゴリ数である．

同様に，e が 1 つの子ノードを持つ場合，すなわち，e が $A_x \to B_y$ であるときは，基底確率を以下のように定義する．

$$P_0(e|A_x) = P_{\mathrm{MLE}}(A \to B) \times \frac{1}{|B|}$$

また，e の子ノードが単語である場合，すなわち，e が $A_x \to w$ であるときは，基底確率を $P_0(e|A_x) = P_{\mathrm{MLE}}(A \to w)$ と定義する．

上記の確率モデルに基づいて基本木 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，e_{i+1} の生成確率は，G_A を積分消去することによって，以下のように計算される．

$$P(e_{i+1}|\mathbf{e}_{1:i}, A_x, d_{A_x}, \theta_{A_x}) = \alpha_{e_{i+1},A_x} + \beta_{A_x} P_0(e_{i+1}|A_x)$$
$$\alpha_{e_{i+1},A_x} = \frac{n_{e_{i+1},A_x} - d_{A_x} \cdot t_{e_{i+1},A_x}}{\theta_{A_x} + \Sigma_e n_{e,A_x}}$$
$$\beta_{A_x} = \frac{\theta_{A_x} + d_{A_x} \cdot \Sigma_e t_{e,A_x}}{\theta_{A_x} + \Sigma_e n_{e,A_x}}$$

ただし，$n_{e_{i+1},A}$ は，$\mathbf{e}_{1:i}$ のうち e_{i+1} と同じ型の基本木が生成された回数を表す．また，$t_{e_{i+1},A}$ は確率モデルの内部変数で，e_{i+1} と同じ型の基本木をいくつのクラスタに分けているかを表す整数である．

シンボル細分化文脈自由文法では，基本木の生成は親ノードのシンボルのみに依存する．したがって，構文木全体の確率は，その構文木を構成する基本木の確率の積となる．すなわち，

$$P(\mathbf{e}) = \prod_{j=1}^{|\mathbf{e}|} P(e_j)$$

である．

2.2　Gibbs サンプリングによる確率文法モデルの学習

一般に，構文木コーパスには文法モデルの基本木の情報は含まれていないため，構文木の各ノード（葉ノードは除く）に潜在変数 z を 1 つずつ割り当て，基本木の情報を表

すことにする．図 3 (b) に，シンボル細分化文脈自由文法の基本木の情報を潜在変数で表したものを示す．シンボル細分化文脈自由文法では，潜在変数 z の値は非終端記号の細分化カテゴリ（$0, 1, \ldots$）を表している．また，木置換文法や木挿入文法などの様々な文法モデルも，同様の方法で基本木の情報を表すことができる [16], [18]．

構文木コーパスから最適な基本木の集合を統計的に推定するには，構文木が与えられたもとでの基本木の事後確率を最大とする潜在変数とパラメータを推定すればよい．

$$\hat{\mathbf{z}}, \hat{\Theta} = \underset{\mathbf{z},\Theta}{\operatorname{argmax}}\, P(\mathbf{z}|\{t\}; \Theta) P(\Theta)$$

ただし，Θ は基本木の確率モデルのパラメータ集合である．

また，推定された潜在変数 $\hat{\mathbf{z}}$ から，基本木の情報を一意に復元することができる．

事後確率を最大化する基本木を求める方法として，Gibbs サンプリング法が広く用いられている．前述のように，Gibbs サンプリング法では，基本木の同時事後分布から直接サンプルを生成するのではなく，各変数を 1 つずつ順番に巡回してサンプリングを行う．アルゴリズムの概要を以下に示す．

(1) 初期潜在変数集合 $\mathbf{z}_0$，初期パラメータ集合 Θ_0 を設定する．
(2) あらかじめ指定された反復回数だけ以下の処理を繰り返す．
  (a) 変数の集合 $\mathbf{z}$ をランダムに並べ替える．
  (b) 並べ替えられた $\mathbf{z}$ を順番に 1 つずつ取り出して，$P(z|\mathbf{z} \setminus z, \Theta)$ の確率で z のサンプルを生成し，潜在変数の値を更新する．
  (c) (b) と同様にパラメータ集合 Θ の値を更新する．

基本木は複数の潜在変数で構成されているため，Gibbs サンプリング法では，基本木を別の基本木へ 1 度に更新することはできない．したがって，ある基本木から別の基本木に至る経路中において非常に確率の低い状態が存在する場合には，基本木はいつまでも同じ状態のままにとどまってしまい，尤度の高い文法規則を効率良く探索できない．また，1 つの基本木に含まれる変数をまとめて同時に更新するブロック化 Gibbs サンプリング法では，上記の問題点は解消できるが，複数の基本木を 1 度に更新することができないため，構文木コーパスの規模が大きくなるにつれて通常の Gibbs サンプリング法と同様の問題が生じる．また，型レベルのブロック化サンプリング法 [9] は，同じ型となる変数をまとめてブロック化し，その中でいくつの変数を反転させるかを確率的にサンプリングする方法である．同じ型の変数とは，着目している変数およびその周囲（親ノードと子ノード）の変数の値が同じである変数の集合をいう．しかしながら，型レベルのブロック化サンプリング法は，Gibbs サンプリングに完全に置き換わる MCMC 法を構成することを目的としており，事後分布に正確に従うマルコフ連鎖を構成するために，ディリクレ過程を事前分布として用いることや，変数の値が二値であることを仮定している[*1]．したがって，そのような制限のために，上記の Pitman-Yor 過程に基づく確率モデルなどには適用することができない．

3. 擬似部分木ブロック化サンプリング法

本章では，我々の提案手法である擬似部分木ブロック化サンプリング法について説明する．我々の狙いは，Gibbs サンプリングのような汎用性を失わずに，なるべく多くの変数を同時にサンプリングする方法を構築することである．そのために，一般的な MCMC 法のステップと，擬似ブロック化サンプリングのステップとを別々に構築して，それらを交互に組み合わせるというアプローチをとる．また，提案手法は，従来のようにどの変数をまとめて同時にサンプリングを行うかという定義を事前に与えるのではなく，サンプリングの反復ごとに適切なブロックを自動的に探索する．したがって，異なる文法モデルにおいても同一のアルゴリズムが適用できるという利点がある．

3.1 部分木のブロック化

提案手法は，各反復計算時において，前回のサンプリング結果である構文木群から，細分化カテゴリまで共通する部分木をまとめてブロック化し，それらに含まれる変数の同時分布からサンプリングを行う．まず，任意の潜在変数の集合 $\mathbf{z} = \{z\}$ に対して，それらが表す部分木を $tree(\mathbf{z})$ とする．たとえば，図 3 (b) で，$\mathbf{z} = \{z_1 = 0, z_2 = 0, z_3 = 1\}$ ならば，$tree(\mathbf{z})$ = (NP-0 (DT-0 NP-1)) である．

ここで，部分木 s に対応する潜在変数の集合 $\mathbf{z}$ のブロック B_s を以下のように定義する．

$$B_s \equiv \{internal(\mathbf{z}) \,|\, tree(\mathbf{z}) = s \wedge \cap \mathbf{z} = \varnothing\} \qquad (1)$$

ただし，$internal(\mathbf{z})$ は，$\mathbf{z}$ に対応する部分木のノードの中で，非終端記号を持つ葉ノードと根ノードに相当する潜在変数を除外したものである．

したがって，ブロック B_s は，各反復時における潜在変数 z の値に依存して決定される．**図 4** に，例として 2 つの構文木を示す．図 4 において，共通の部分木 s = (A-0 (B-0 (C-1 (D-2 E-0)))) に対応するブロックは，$B_s = \{\{z_2, z_3\}, \{z_{11}, z_{12}\}\}$ となる．A-0，D-2，E-0 は，部分木 s の中で非終端記号を持つ葉ノードまたは根ノードであり，これらに対応する変数の値を変更してしまうと，その周囲の基本木（たとえば，(G-1 (A-0 K-0))）の情報も同時に変更してしまうことになるためブロックから除外する．B-0 は部分木 s の葉ノードであるが，前終端記号を持つので，子ノードは必ず構文木の葉ノードである．したがって，

[*1] 多値変数である場合は，サンプリングの反復ごとに，スライスサンプリング [13] などの方法で二値に変換する．

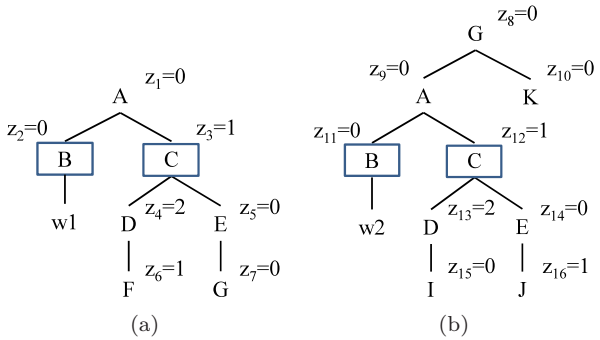


図 4　2 つの構文木と，部分木のブロックの例
Fig. 4　Example parse trees and blocked subtrees.

B-0 に対応する変数の値を変更しても周囲の基本木に影響を与えないため，ブロックに含める．

ブロックを構築した後に，以下の同時確率に従って部分木のサンプリングを行う．

$$P\left(\{\mathbf{z}\}_{\mathbf{z}\in B_s}|\mathbf{z}^-,\Theta\right) \tag{2}$$

ただし，$\{\mathbf{z}\}_{\mathbf{z}\in B_s}$ は，B_s に含まれるすべての変数の集合を表し，$\mathbf{z}^-$ は，構文木コーパス中のすべての変数から B_s に含まれる変数を取り除いた集合である．

z の値が c 通りの可能性を持つとき，式 (2) の変数の値の組合せは，$c^{|\mathbf{z}|\times|B_s|}$ 通りである．$|B_s|$ は構文木コーパスのデータ量に応じて増大するため，すべての変数 z の値の組合せについて式 (2) を求めることは計算量的に困難である．そこで，ブロック内で共通部分木の同じノードに対応する潜在変数は，サンプリング後も必ず同じ値になるという制約を設ける．このようにすると，ブロックに含まれる変数の集合 $\mathbf{z}\in B_s$ を 1 つだけ取り出して，それらの値のとりうる可能性のみを考慮すればよい．したがって，組合せの数は $c^{|\mathbf{z}|}$ 通りになる．図 4 の例では，z が二値変数であるとすると，$(z_2,z_3,z_{11},z_{12})=(0,0,0,0),(0,0,0,1),\ldots(1,1,1,0),(1,1,1,1)$ の 16 通りすべての可能性について式 (2) を計算するのではなく，$(0,0,0,0)$，$(0,1,0,1)$，$(1,0,1,0)$，$(1,1,1,1)$ の 4 通りのみについて式 (2) を計算してサンプリングを行う（擬似サンプリング）．

潜在変数の値に依存したブロック化を利用すること，また，上記の制約を導入して計算量を削減することの代償として，式 (2) に基づくサンプリングは，基本木の事後分布に従うサンプルを生成しない．そこで，Gibbs サンプリング法などの一般的な MCMC 法と，上記の擬似部分木ブロック化サンプリング法とを組み合わせて使用することを提案する．以下に，ブロック B_s の構築方法と，提案手法の具体的なアルゴリズムについて述べる．

3.2　ブロックの構築

ブロック B_s を構築するために，サンプリングの反復ごとに，構文木コーパスから潜在変数を考慮した共通の部分木を探索する必要がある．我々は，パターンマイニングの手法に基づいて共通の部分木を列挙することを提案する．具体的な手順は以下のとおりである．まず，1 つのノードのみからなる最小の部分木から始めて，そこに子ノードを追加して部分木を拡大する．この手続きを再帰的に繰り返すと，部分木パターンをノードとする木構造が生成される．部分木の拡大は，探索中の部分木パターンがあらかじめ設定した最大ノード数に達するか，または頻度が 1 になったときに停止する．これは，FREQT [1] に代表される木構造のパターンマイニング手法と本質的に同等である．部分木パターンの集合を発見した後，構文木コーパスの全ノードが必ずどこか 1 つのブロックに所属するまでランダムに部分木パターンを 1 つ選択し，そこからブロック B_s を構築する．上記の手続きを行うことによって，すべてのノードが 1 度ずつ含まれたブロックの集合 $\mathbf{B}=\{B_s\}$ が構築できる．また，実装上の工夫として，細分化情報を除いた生の構文木コーパスからあらかじめ共通の部分木パターンを列挙しておき，その範囲内で上記の部分木パターンマイニングを行うことで計算の無駄を省くことができる．

3.3　提案手法のアルゴリズム

前述のように，提案手法は，一般的な MCMC 法と擬似部分木ブロック化サンプリング法とを組み合わせて使用する．提案手法の具体的な手続きをアルゴリズム 1 に示す．アルゴリズム 1 の入力は，サンプリングの反復回数 I，構文木コーパス $\{t\}$，擬似ブロック化サンプリングの頻度 f である．頻度 f については後述する．

アルゴリズム 1 では，まず，通常の Gibbs サンプリング法によって変数の値を更新する（行 4）．Gibbs サンプリング法以外の任意の MCMC 法を用いてもよい．次に，現在の反復値 i が，あらかじめ設定された擬似部分木ブロック化サンプリング法の頻度 f の条件を満たすならば，擬似ブロック化サンプリングを実行する．たとえば，$f=10$ とすると，10 回の Gibbs サンプリングを行うたびに 1 度だけ擬似ブロック化サンプリングを実行する．頻度 f を導入する理由は，擬似部分木ブロック化サンプリングの実行に要する計算コストと，探索効率とのバランスを調整できるようにするためである．頻度 f が文法規則の探索効率に与える影響は実験で評価する．

擬似部分木ブロック化サンプリング法では，まずはじめに，部分木の段階的な拡大によるパターンマイニング手法によって，共通の部分木を探索する（行 9）．次に，変数 Z と $\mathbf{B}$ を用意して，ブロックの構築と，すべての潜在変数 z が必ずどこかのブロックに所属するような割当てを行う．Z は，確率文法モデルの潜在変数 z がどこかのブロック B_s に所属したかを管理し，$\mathbf{B}$ は，構築されたブロック B_s を格納する．具体的な手続きは，まず，部分木パターンをランダムに選択してブロック B_s を逐次的に構築し，$\mathbf{B}$

Algorithm 1: 部分木ブロック化サンプリング法

Input : number of iterations: I, parse trees: $\{t\}$, frequency of blocked subtree sampling: f
Output: estimated elementary trees: $\hat{\mathbf{e}}$, estimated parameters: $\hat{\Theta}$

1 **for** $i = 1, \ldots, I$ **do**
2 　Initialize $\mathbf{z}, \Theta$
　 `// Gibbs sampling`
3 　**foreach** z in random order **do**
4 　　Generate z' according to $P(z|\mathbf{z} \setminus z, \Theta)$
5 　　$z \leftarrow z'$
6 　**end**
7 　Update parameters Θ
8 　**if** i mod $f = 0$ **then**
　　 `// Construct block`
9 　　Find subtree patterns S by subtree expansion method
10 　　$Z \leftarrow \mathbf{z}$
11 　　$\mathbf{B} \leftarrow \emptyset$
12 　　**while** $Z \neq \emptyset$ **do**
13 　　　Pick subtree $s \in S$ at random
14 　　　Construct B_s
15 　　　$\mathbf{B} \leftarrow \mathbf{B} \cup B_s$
16 　　　**foreach** z in B_s **do**
17 　　　　$Z \leftarrow Z \setminus z$
18 　　　**end**
19 　　**end**
　　 `// Pseudo blocked subtree sampling`
20 　　**foreach** $B_s \in \mathbf{B}$ in random order **do**
21 　　　Generate $\{\mathbf{z}\}'$ according to $P\left(\{\mathbf{z}\}_{\mathbf{z} \in B_s} | \mathbf{z}^-, \Theta\right)$
22 　　　$\{\mathbf{z}\} \leftarrow \{\mathbf{z}\}'$
23 　　**end**
24 　**end**
25 **end**
26 Recover $\hat{\mathbf{e}}$ from $\hat{\mathbf{z}}$

に追加していく（行 15）．次に，構築されたブロックに含まれる潜在変数を Z から削除する（行 17）．構文木コーパスに含まれるすべての潜在変数がいずれかのブロックに所属したときに $Z = \emptyset$ となり，割当ては完了する．

ブロックの構築と潜在変数の割当てが完了したら，実際に擬似ブロック化サンプリングを行う．具体的には，ブロックの集合 $\mathbf{B}$ からランダムに 1 つのブロックを選択し，そのブロックに含まれる潜在変数の値の組合せについて式 (2) を計算し，その確率に従ってサンプルを生成する（行 21）．以上の手続きを指定された反復数だけ繰り返し，潜在変数の値を更新していく．最後に，反復が終了したら，その時点における潜在変数の推定値 $\hat{\mathbf{z}}$ から基本木の情報 $\hat{\mathbf{e}}$ を復元して，アルゴリズム 1 は終了する（行 26）．

4. 実験

4.1 設定

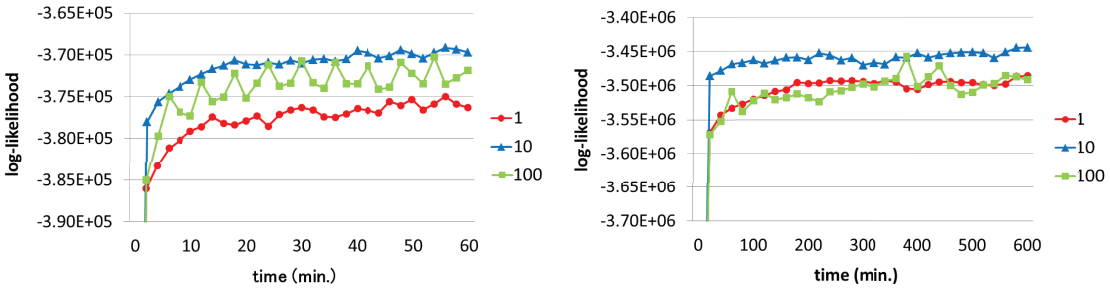
英語の構文木コーパスである WSJ Penn Treebank [11] を用いて，提案手法の評価を行った．Penn Treebank データはセクション単位で区切られており，各セクションは約 2,000 文の構文木で構成されている．本実験では，データ量の違いが各手法に与える影響を評価するため，Penn-A データ（セクション 2 のみ，1,989 文）と，Penn-B データ（セクション 2 から 11 まで，18,581 文）の 2 種類のデータセットを用いた．データの前処理として，コーパス中の全構文木を “Right-Binarized” 法 [12] によって二分木へ変換した．また，コーパス中に 1 度しか現れない単語は，“UNKNOWN” に置き換えた．実験に用いる確率文法モデルは，2 章で説明した Pitman-Yor 過程に基づく確率シンボル細分化文脈自由文法である．サンプリングの反復数は，既存研究を参考にして 5,000 回とした [17]．これは，確率文法モデルの学習結果を構文解析器で利用する際に十分な反復数である．また，実験結果で示す対数尤度は，各手法をそれぞれ独立に 5 回試行したときの平均である．使用した計算機は，CPU が Core i7 3.07 GHz，メモリが 18 GB である．

4.2 結果

4.2.1 擬似部分木ブロック化サンプリングの頻度の比較

まずはじめに，提案手法のアルゴリズム 1 において，擬似部分木ブロック化サンプリングを行う頻度 f を変化させたときの探索効率の違いを評価した．実験設定として，細分化のカテゴリ数は 2 とし，潜在変数の初期値はランダムに決定した．**図 5** に，擬似部分木ブロック化サンプリングの頻度を 1，10，100 と変化させたときの対数尤度の実験結果をグラフで示す．擬似部分木ブロック化サンプリングの頻度によって変数の更新回数が異なるため，横軸はサンプリングの反復数ではなく，時間（分）で示してある．

図 5 に示されているように，擬似部分木ブロック化サンプリングの頻度によって，尤度の高い解に到達するまでの時間に違いが見られた．これは，擬似部分木ブロック化サンプリングは Gibbs サンプリングよりも計算コストが高いため，Gibbs サンプリングと擬似ブロック化サンプリングを 1 度ずつ交互に繰り返すよりも，10 回または 100 回ごとに実行したほうが探索効率が良い結果であったと考えられる．また，Penn-A データセットと Penn-B データセットの結果を比較すると，いずれも頻度 10 のときが最も良い結果であるが，Penn-A データセットにおいて頻度 100 は頻度 1 よりも探索効率が高いのに対して，Penn-B データセットでは，双方にあまり差が見られない．これは，データ量が増大することによって擬似部分木ブロック化サンプリングの効果が相対的に高くなったため，小規模データの



(a)　(b)

図 5　擬似部分木ブロック化サンプリングの頻度の比較．(a) Penn-A データセットでの結果，(b) Penn-B データセットでの結果

Fig. 5　Frequency comparison of the pseudo blocked subtree sampler. (a) Results on Penn-A data. (b) Results on Penn-B data.

(a)　(b)

図 6　細分化カテゴリを 2 としたときの他手法との比較．(a) Penn-A データセットでの結果，(b) Penn-B データセットでの結果

Fig. 6　Comparison with other methods when the number of symbol subcategories is 2. (a) Results on Penn-A data. (b) Results on Penn-B data.

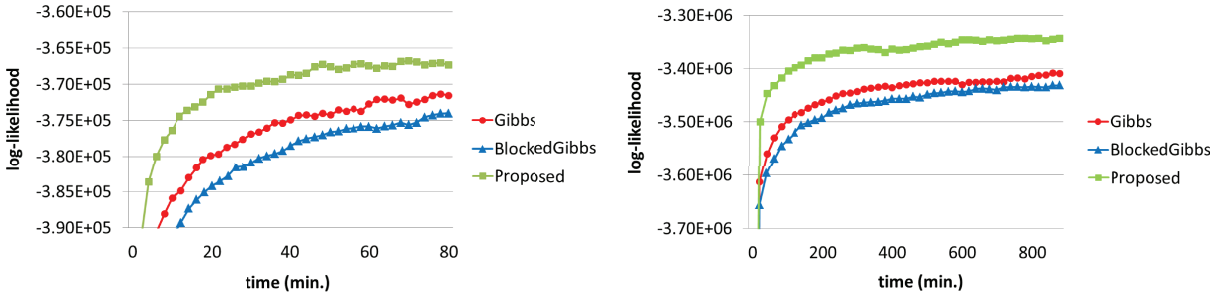
場合は，頻繁に擬似ブロック化サンプリングを行うよりも計算コストの低い Gibbs サンプリングを何度も行ったほうが探索効率が良かったのに対して，中規模データでは，計算コストをある程度要してでも擬似ブロック化サンプリングを頻繁に行ったほうが探索効率が良いという結果になったと考えられる．

4.2.2　他手法との比較

次に，提案手法と既存手法との比較実験を行った．既存手法として，Gibbs サンプリング法とブロック化 Gibbs サンプリング法を用いた．ブロック化 Gibbs サンプリング法は，潜在変数を 1 つずつ巡回してサンプリングを行うのではなく，基本木を表す複数の潜在変数をまとめてサンプリングを行う方法である．たとえば，図 3 (b) では，潜在変数を $B_1 = \{z_1, z_2, z_3\}$，$B_2 = \{z_4\}$，$B_3 = \{z_5\}$ の 3 つのブロックに分けて，それぞれのブロックに対して式 (2) を計算し，サンプリングを行う．**図 6** に，提案手法と既存手法の実験結果をグラフで示す．細分化のカテゴリ数は 2 に設定し，潜在変数の初期値はランダムに決定した．また，提案手法における擬似部分木ブロック化サンプリングの頻度は 10 に設定した．

図 6 に示されているように，提案手法は，既存手法と比較して最も探索効率が良い手法であった．Penn-A データを用いた場合，ブロック化 Gibbs サンプリング法は複数の変数をまとめて更新しているにもかかわらず，通常の Gibbs サンプリング法よりも尤度の高い解に到達するのに余計に時間がかかるという結果であった．これは，小規模データでは，ブロック化された変数の同時確率を計算するために時間を多く使うよりも，少ない計算量で変数の更新回数を多くするほうが良い場合があることを示している．ただし，図 6 (a) において，ブロック化 Gibbs サンプリング法と通常の Gibbs サンプリング法との対数尤度は徐々に縮まっていき，計算時間が 60 分（Gibbs サンプリング法の反復数が約 5,000 回に達したとき）にはほぼ同じ値に到達した．これは，Gibbs サンプリング法が 30 分を超えたあたりから局所解にとどまってしまい，それ以上尤度の高い解へ到達できない状態へ陥っているのに対し，ブロック化 Gibbs サンプリング法では，基本木を 1 度に別の基本木へ更新できるため，そのような問題が生じにくいためであると考えられる．一方，提案手法は，ブロック化 Gibbs サンプリング法と比較して，1 つの基本木ではなく，構文木コーパスにまたがる複数の部分木をブロック単位として同時にサンプリングを行うことができるため，短時間で尤度の高い解へ到達することができる．

Penn-B データを用いた場合には，ブロック化 Gibbs



(a)　(b)

図 7　細分化カテゴリを 3 としたときの他手法との比較．(a) Penn-A データセットでの結果，(b) Penn-B データセットでの結果

Fig. 7　Comparison with other methods when the number of symbol subcategories is 3. (a) Results on Penn-A data. (b) Results on Penn-B data.

サンプリング法と通常の Gibbs サンプリング法はほぼ同じ結果であった．Penn-A データと比較すると，データ量が増大することによって，単純な Gibbs サンプリング法ではますます尤度の高い解へ到達することが困難となり，ブロック化 Gibbs サンプリング法の優位性が相対的に向上していると考えられる．提案手法は，他手法と比較してきわめて良い性能である．特に，サンプリングの反復数が少なく尤度の低い状態のときに，提案手法では構文木全体にまたがる複数の変数をまとめて更新することによって尤度の高い状態へ短時間で到達していることが確認できる．

次に，細分化カテゴリの数を 3 と設定したときの結果を**図 7** に示す．先ほどの実験と同様に，潜在変数の初期値はランダムに決定し，擬似部分木ブロック化サンプリングの頻度は 10 に設定した．細分化カテゴリを増やしても，提案手法は他手法と比較して最も探索効率の良い手法であった．これは，前述のように，構文木コーパスにまたがる複数の基本木を 1 度に別の基本木へ更新することができるため，他手法に比べて局所的な状態にとどまりにくいためであると考えられる．細分化カテゴリの数を 2 から 3 へ増加させると，サンプリングに要する計算コストは $1.5^{|\mathbf{z}|}$ 倍になる．ただし，$|\mathbf{z}|$ は，1 度にサンプリングを行う潜在変数の数である．したがって，1 度に多くの潜在変数を更新する方法ほど，計算量の増分は大きくなる．以上の要因から，細分化カテゴリの数が 2 のときと比較して，ブロック化 Gibbs サンプリング法よりも Gibbs サンプリング法の探索効率が良くなっていると考えられる．一方，提案手法も計算量は増加しているが，擬似ブロック化サンプリングを毎回実行するのではなく，適当な頻度 f で実行することによってサンプリングの計算コストを削減しているため，細分化カテゴリの数を増やしても，他手法に比べて効率が良いという結果になったと考えられる．

5.　関連研究

テーブルラベル再サンプリング法 [8] は，Adaptor 文法の学習に特化したブロック化サンプリング法であり，Gibbs サンプリング法などの MCMC 法と併用して用いるという点において本手法と共通している．ただし，Adaptor 文法は，基本木の葉ノードが必ず終端記号（単語）でなければならないという制約があるため，シンボル細分化文脈自由文法や木置換文法にテーブルラベル再サンプリング法を直接適用することはできない．また，Adaptor 文法は，主に構文木コーパスを前提としない完全な教師なし学習に基づくタスク（たとえば教師なし単語分割）に使用され，木構造ではなくシーケンスに対するブロック化サンプリングであるといえる．

型レベルのブロック化サンプリング法 [9] は，同じ型となる変数をまとめてブロック化し，その中でいくつの変数を反転させるかを確率的にサンプリングする方法である．同じ型の変数とは，着目している変数およびその周囲（親ノードと子ノード）の変数の値が同じである変数の集合のことである．したがって，我々の手法のように部分木をブロックとするのではなく，変数をブロック化する方法であるという違いがある．また，型レベルのブロック化サンプリング法は，Gibbs サンプリングに完全に置き換わる MCMC 法を構成することを目的としており，事後分布に正確に従うマルコフ連鎖を構成するために，ディリクレ過程を事前分布として用いることや，変数の値が二値であることを仮定している．したがって，そのような制限のために，上記の Pitman-Yor 過程に基づく確率モデルなどには適用することができない．

6.　おわりに

本稿では，統計的文法獲得において Gibbs サンプリング法が局所最適解にとどまりやすく尤度の高い解を効率的に探索できないという問題を改善するために，部分木を単位とする新たな擬似ブロック化サンプリング法を提案した．提案手法は，パターンマイニングの手法に基づいて適切なブロックを自動的に獲得し，それらをまとめて同時にサンプリングを行う．また，同じ部分木はサンプリング後も同じ変数の値になるという制約を設けて計算量を削減する代

わりに，通常の MCMC 法と擬似部分木ブロック化サンプリング法とを組み合わせて使用する．ただし，この制約によって，文法規則の近似事後分布からのサンプリングとなる．確率シンボル細分化文脈自由文法を用いて提案手法の評価を行った結果，本手法は，既存手法よりも探索効率が高く，尤度の高い文法規則が獲得できることを確認した．今後は，提案手法を変分ベイズ法と比較することや，様々な文法モデルにおける本手法の有効性を検証する必要がある．

参考文献

[1] Abe, K., Kawasoe, S., Asai, T., Arimura, H. and Arikawa, S.: Optimized substructure discovery for semi-structured data, *Proc. 6th European Conference on Principles and Practice of Knowledge Discovery in Databases* (*PKDD*), pp.1–14 (2002).

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

[3] Cohn, T., Blunsom, P. and Goldwater, S.: Inducing Tree-Substitution Grammars, *Journal of Machine Learning Research*, Vol.11, pp.3053–3096 (2010).

[4] 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).

[5] Cohn, T. and Lapata, M.: Sentence Compression Beyond Word Deletion, *Proc. 22nd International Conference on Computational Linguistics*, pp.137–144, Association for Computational Linguistics (2008).

[6] Galley, M., Hopkins, M., Knight, K. and Marcu, D.: What's in a Translation Rule, *Proc. Human Language Technologies: The 2004 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (*NAACL/HLT*), Vol.4, pp.273–280 (2004).

[7] Geman, S. and Geman, D.: Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images, *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol.PAMI-6, pp.721–741 (1984).

[8] 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).

[9] Liang, P., Jordan, M.I. and Klein, D.: Type-Based MCMC, *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Los Angeles, California, pp.573–581, Association for Computational Linguistics (2010).

[10] Liang, P., Petrov, S., Jordan, M.I. and Klein, D.: The infinite PCFG using hierarchical Dirichlet processes, *Proc. 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning* (*EMNLP-CoNLL*), pp.688–697 (2007).

[11] Marcus, M.P., Santorini, B. and Marcinkiewicz, M.A.: Building a Large Annotated Corpus of English: The Penn Treebank, *Computational Linguistics*, Vol.19, pp.313–330 (1993).

[12] 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).

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

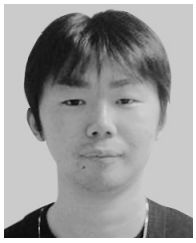
[14] Petrov, S., Barrett, L., Thibaux, R. and Klein, D.: Learning Accurate, Compact, and Interpretable Tree Annotation, *Proc. 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics* (*ICCL-ACL*), Sydney, Australia, pp.433–440,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

[15] 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).

[16] Shindo, H., Fujino, A. and Nagata, M.: Insertion Operator for Bayesian Tree Substitution Grammars, *Proc. 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, Oregon, USA, pp.206–211, Association for Computational Linguistics (2011).

[17] Shindo, H., Miyao, Y., Fujino, A. and Nagata, M.: Bayesian Symbol-Refined Tree Substitution Grammars for Syntactic Parsing, *Proc. 50th Annual Meeting of the Association for Computational Linguistics*, Jeju Island, Korea, pp.440–448, Association for Computational Linguistics (2012).

[18] Yamangil, E. and Shieber, S.: Estimating Compact Yet Rich Tree Insertion Grammars, *Proc. 50th Annual Meeting of the Association for Computational Linguistics* (*Volume 2: Short Papers*), Jeju Island, Korea, pp.110–114, Association for Computational Linguistics (2012).



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（2012 年），情報処理学会コンピュータサイエンス領域奨励賞（2013 年）等受賞．ACL 会員．

松本 裕治　（フェロー）

1977 年京都大学工学部情報工学科卒業．1979 年同大学大学院工学研究科修士課程情報工学専攻修了．同年電子技術総合研究所入所．1984～1985 年英国インペリアルカレッジ客員研究員．1985～1987 年（財）新世代コンピュータ技術開発機構に出向．京都大学助教授を経て，1993 年より奈良先端科学技術大学院大学教授，現在に至る．工学博士．専門は自然言語処理．人工知能学会，言語処理学会，認知科学会，AAAI，ACL，ACM 各会員．ACL Fellow．

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．