階層型隠れマルコフモデルの高速パラメタ推定

若林　啓†　三浦　孝夫†

† 法政大学 工学研究科　〒 184–8584 東京都小金井市梶野町 3-7-2
E-mail: †kei.wakabayashi@gs-eng.hosei.ac.jp, ††miurat@k.hosei.ac.jp

あらまし　本研究では，階層型隠れマルコフモデル (HHMM) の新しいパラメタ推定アルゴリズムを提案する．HHMM のパラメタ推定のアルゴリズムにはいくつかの方法が提案されているが，階層の深さや状態数に対する計算量が大きいことが問題であった．本研究では，HHMM の確率変数を変換することで，HHMM の EM アルゴリズムによるパラメタ推定が，状態の活性化確率についての Forward-Backward アルゴリズムとして効率的に計算できることを示す．
キーワード　階層型隠れマルコフモデル，動的ベイジアンネットワーク，Forward-Backward アルゴリズム

Efficient Parameter Estimation for Hierarchical Hidden Markov Models

Kei WAKABAYASHI† and Takao MIURA†

† Dept.of Elect.& Elect. Engr., HOSEI University　3-7-2, KajinoCho, Koganei, Tokyo, 184–8584 Japan
E-mail: †kei.wakabayashi@gs-eng.hosei.ac.jp, ††miurat@k.hosei.ac.jp

Abstract　In this work, we propose a new parameter estimation algorithm for Hierarchical Hidden Markov Models (HHMM). There are several methods for HHMM parameter estimation so far, but computation complexities of the conventional methods are impractical for HHMMs which has the large hierarchical depth or the large number of states. In this paper, we show that the strict EM algorithm for HHMM can be executed efficiently as the Forward-Backward algorithm for state activation probabilities.
Key words　Hierarchical Hidden Markov Model, Dynamic Bayesian Network, Forward-Backward Algorithm

1. ま え が き

確率モデルを用いた機械学習は，様々な分野に適用できる汎用性，不確定なデータに対する適応性といった点で，近年特に注目されてきている．隠れマルコフモデル (Hidden Markov Model; HMM) は系列データを扱う確率過程モデルであり，系列を成す要素それぞれに対して状態を表す１つの潜在確率変数を仮定し，潜在変数のマルコフ過程として系列データをモデル化する．音声認識，単語の品詞推定，楽曲のコード推定など様々な問題が，HMM の潜在変数を推定する問題として扱われる．

HMM が多様な問題に適用される中で，HMM にいくつかの欠点が指摘され，そのためのいくつかの拡張モデルが提案されている [2], [6], [11], [14]．その中でも，HMM は複数の観測値が１つの句を成すような系列においてパターンを抽出するのが難しいという指摘がある．例えば，固有名詞抽出では人名や地名，時間表現といった固有表現クラスを推定することが目的であるが，これらの固有表現は複数の単語から構成されるため，１つの潜在変数が１つの単語にしか対応しない HMM では自然なモデル化が難しい．この問題に対して Bikel ら [1] は，HMM の状態と単語マルコフモデルを１対１で対応させ，１つの状態が当該マルコフモデルを用いて複数の単語を続けて出力する手法を提案している．これは，固有表現クラスのマルコフ過程と，単語のマルコフ過程の２階層を考えることに対応する．このことから，階層的な隠れマルコフモデルを用いることで，複数の観測値が１つの句を成すような系列パターンをうまく捉えられることを示唆している．

階層型隠れマルコフモデル (Hierarchical Hidden Markov Model; HHMM) は，HMM に階層的な出力を許す確率過程モデルである [5]．HHMM は，上位階層の状態が下位階層の部分 HMM を再帰的かつ確率的に出力することで，階層的に系列データを生成する．下位階層の部分 HMM は，終了状態と呼ばれる特殊な状態に遷移することができ，このとき初めて上位階層の状態遷移が起こる．部分 HMM はモデルのパラメタに従って尤もらしい出力を生成するから，部分 HMM が出力されてから終了するまでの部分系列をまとまった句として考えることができる．HHMM を用いることで，文字列の単語，文節構造 [5] や，固有表現の句構造 [10]，楽曲のフレーズ構造 [12]，行動パターンの階層構造 [7], [9] などを扱うことができる．

本稿では，HHMM のパラメタ推定アルゴリズムを提案する．従来の HHMM のパラメタ推定アルゴリズムは，HHMM の状態数や階層の深さに対する計算量が大きく，大規模な状態空間を持つ HHMM のパラメタ推定は困難であった．本研究では，

従来知られているパラメタ推定手法よりも状態数や階層の深さに対する計算量が小さいパラメタ推定手法を提案する．本手法により，より大規模な状態空間を持つ HHMM のパラメタ推定が高速に実行可能となる．

本稿の構成は以下の通りである．２章で関連研究について述べ，本研究の位置づけを明らかにする．３章で階層型隠れマルコフモデルについて述べ，４章で提案するパラメタ推定アルゴリズムについて論じる．５章で実験結果を示し，提案アルゴリズムが高速に実行できることを示す．６章で結論を述べる．

2. 関 連 研 究

HHMM を提案した Fine ら [5] は，一般化 Baum-Welch アルゴリズムと呼ばれるパラメタ推定手法を述べている．この手法は，部分 HMM の開始時刻と終了時刻(注1)を指定した上での状態の滞在確率を求めることで，状態の遷移回数の期待値を求める．一般化 Baum-Welch アルゴリズムは，全ての状態数を S，学習データの系列長を T とすると，時間計算量は $O(ST^3)$ を要する．この計算量は，直観的には，部分 HMM の開始時刻の選択で T 通り，終了時刻の選択で最悪 T 通り，その時刻間の状態の滞在確率の計算に最悪 T の計算量を要するため，各部分 HMM につき計算量が T^3 となることによる．このため，系列長が大きくなると爆発的に実行時間が増加して，現実的な時間では実行することができない．

Murphy ら [8] らは，理論的な計算量が求められないが，実験的には系列長に対して線形な計算量で実行できる HHMM のパラメタ推定手法を示した．ここでは，HHMM が等価な動的ベイジアンネットワーク (Dynamic Bayesian Network; DBN) で表現できることを示し，任意の DBN に対して効率的な期待値計算を行うジャンクションツリーアルゴリズムを適用することでパラメタ推定を実現している．Murphy らの大きな貢献は，図 1 に示すように，各階層の状態と，各階層の終了フラグを確率変数として持つことで，ある時刻の確率変数が直前の時刻の確率変数にのみ依存することを示した点にある．すなわち，一般化 Baum-Welch アルゴリズムのように部分 HMM の開始時刻と終了時刻の全てのパターンを列挙しなくても，各時刻での部分 HMM 終了フラグの期待値を計算することで，状態の確率について等価な期待値を得ることができることを意味する．

ジャンクションツリーアルゴリズムは，ベイジアンネットワークを表現するグラフ構造に対して機械的な操作を繰り返すことにより，ジャンクションツリーと呼ばれる効率的な確率伝播ツリー構造を推定する手法である [4]．この特性により，ジャンクションツリーアルゴリズムは任意のベイジアンネットワークの確率変数の期待値計算に適用できるが，ツリーの生成方法が一意でないため，理論的な計算量の見積もりが難しい．Murphy らは，実験的に，HHMM のジャンクションツリーアルゴリズムによるパラメタ推定の時間計算量の上限を，階層の深さを D，１階層の状態数を N としたとき $O(T(D+2)N^{1.5D}2^{0.5D})$ としている．しかし，これはあくまで実験的な計算量であり，理論的な計算量を求められるアルゴリズムが必要である．また，ジャンクションツリーアルゴリズムは，一般的なベイジアンネットワークに適用できる汎用アルゴリズムのため，実装が複雑であるという問題がある．

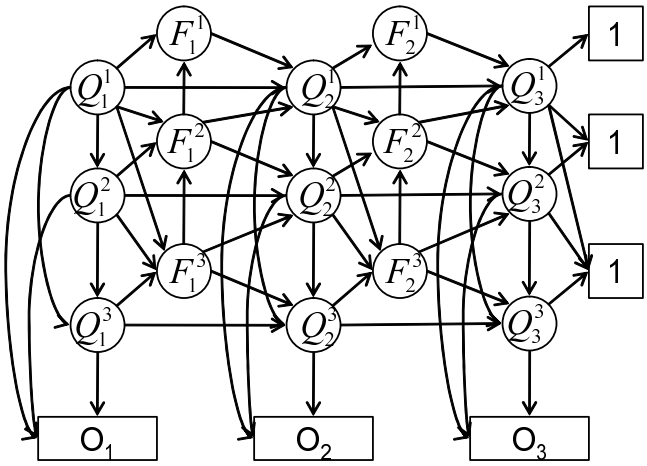

図 1　HHMM の DBN 表現

HHMM が等価な DBN に変換できることから，HHMM の隠れ状態は，全ての階層の状態の組み合わせのマルコフ過程に従う．このため，全ての階層の状態の組み合わせを１つの離散値の内部状態で表現することで，HHMM を等価な HMM に変換できる．この等価な HMM に対して HMM のパラメタ推定手法を適用することで，緩い条件の下で HHMM のパラメタ推定が実現できる [12], [13]．このとき，変換した HMM の状態数は N^D であることから，パラメタ推定の時間計算量は $O(TN^{2D})$ である．この計算量は理論的に求められ，学習データの系列長 T に対して線形の特性を持つが，Murphy らの手法によって実験的に得られた計算量よりも大きく，状態数 N や階層の深さ D が大きいとき効率的でない．

本研究では，時間計算量 $O(TN^{D+1})$ で実行できる HHMM のパラメタ推定アルゴリズムを提案する．本アルゴリズムでは，HMM のパラメタ推定を効率的に行う Forward-Backward アルゴリズムに基づいてパラメタ推定を行う．HHMM では，時刻の経過に対して状態の遷移を必ずしも伴わないため，素直に Forward-Backward アルゴリズムを適用することはできない．ここでは，状態の遷移が起こる確率を状態の活性化確率と呼び，DBN の確率変数の同時確率を用いて定義する．本稿では，それぞれの状態の周辺確率が，状態の活性化確率についての Foward-Backward アルゴリズムとして計算できることを示し，この確率に基づいて効率的にパラメタ推定が実行できることを示す．また，本アルゴリズムの導出は EM アルゴリズムに基づいているため，パラメタの収束が保証され，その計算量を求めることができることを示す．

3. 階層型隠れマルコフモデル

図 1 は，深さ 3 の階層型隠れマルコフモデル (HHMM) の動的ベイジアンネットワーク (DBN) 表現である．図中の O_t が

(注1)：本稿で時刻とは，単に系列データの要素に対するインデックスを意味する．

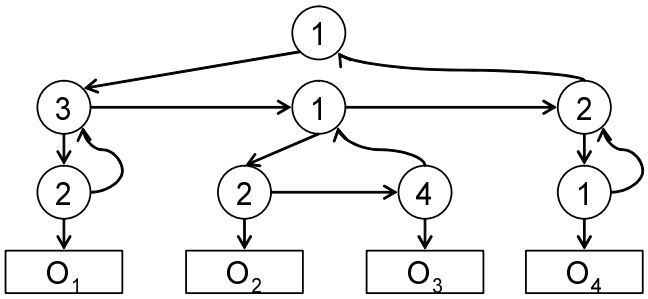

図 2　HHMM が系列を生成する過程

系列データを構成する観測値であり，下付き文字 t は系列における時刻を表す．ここで時刻とは，系列の始めから順に与えられた整数のインデックスである．Q_t^d は隠れ状態であり，下付き文字 t が対応する時刻，上付き文字 d が対応する階層を表す．ここでは深さ 3 の HHMM を考えているため，各時刻に対応して 3 つの隠れ状態があり，最上層が $d=1$，最下層が $d=3$ としてインデックスを与える．F_t^d は，時刻 t で階層 d の状態が終了状態に遷移したかどうかを表す 2 値の確率変数である．

図 2 に，深さ 3 の HHMM が系列を生成する過程の例を示す．ここでは，図中の直線矢印は状態の生成を示しており，曲線矢印は制御の移動を示している．系列の開始において，HHMM は時刻 1 の最上層の状態を確率的に生成する．状態は離散値の確率変数であり，ここでは最上層の状態として $Q_1^1=1$ が生成されている．次に，最上層の状態に依存した確率分布に従って，階層 2 の状態 $Q_1^2=3$ を生成する．さらに，階層 1 の状態と階層 2 の状態に依存した確率分布に従って，階層 3 の状態 $Q_1^3=2$ を生成する．このように，下位階層の状態はその階層より上の全ての階層の状態に依存して生成され，この上から下に向かう遷移の確率を HHMM の初期状態確率と呼ぶ．

最下層の状態まで生成したら，全ての階層の状態に依存した確率分布に従って観測値 O_1 を生成する．この観測値を生成する確率を HHMM の出力確率と呼ぶ．

時刻が 1 進むと，最下層の状態が上位の全ての階層の状態に依存した確率分布に従って状態遷移を行う．この横方向の状態遷移の確率を状態遷移確率と呼ぶ．ただし遷移先にはただ 1 つの「終了状態」が存在しており，終了状態に遷移した場合，1 つ上の階層に制御が移る．この例の場合，時刻 1 で階層 3 の状態が終了状態に遷移し，階層 2 で状態遷移が起こっている．階層 2 での状態遷移も同様に上位の全ての階層の状態に依存した遷移確率分布に従って状態遷移が起こり，ここでは状態 $Q_2^2=1$ に遷移している．階層 2 で終了状態に遷移しなかったので，それより上の階層の状態は時刻 1 の状態から変化しない．また，階層 2 で遷移が起きたため，階層 3 の状態は上位の全ての階層の状態に依存した初期状態確率に従って生成される．

この過程を最終時刻まで繰り返し，HHMM は観測値の系列を生成する．最終時刻では，最上層を含む全ての階層の状態は終了状態に遷移する．図 1 において，最終時刻の終了フラグ F_3^d の代わりに 1 が与えられているのは，この制約のためである．

上位の階層が異なると，同じ状態のインデックスでも異なる状態を表すことに注意したい．例えば，図 2 で，時刻 1 の階層 3 の状態 Q_1^3 と，時刻 2 の階層 3 の状態 Q_2^3 は，共に 2 というインデックスで示してあるが，状態遷移確率，出力確率，初期状態確率はいずれも上位の全ての階層の状態に依存して確率分布が与えられるという特性から，上位の階層の状態が異なる場合，インデックスが同値であることに意味はない．

表 1　記号の定義

D	階層の深さ
N	1 階層の状態数
$Q_t^d \in 1,...,N$	時刻 t における階層 d の状態
$Q_t^{1:d} \in 1,...,N^d$	時刻 t における最上層から階層 d までの状態の組み合わせ
$F_t^d \in 0,1$	時刻 t における階層 d の終了フラグ
$O_t \in 1,...,K$	時刻 t の観測値
$A_m^d(i,j)$	$Q_t^{1:d-1}=m$ のときの階層 d における状態 i から状態 j への遷移確率
$A_m^d(i,end)$	$Q_t^{1:d-1}=m$ のときの階層 d における状態 i から終了状態への遷移確率
$\pi_m^d(i)$	$Q_t^{1:d-1}=m$ のときの階層 d における状態 i の初期状態確率
$B_m(k)$	$Q_t^{1:D}=m$ のときの観測値 k の出力確率

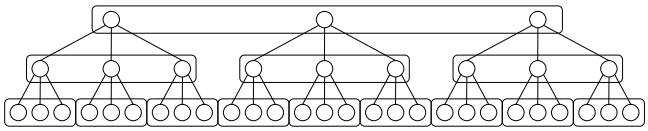

図 3　HHMM の状態空間の木構造による表現

HHMM における記号の定義を表 1 に示す．HHMM では，状態遷移確率，初期状態確率，出力確率はいずれも上位階層の状態の組み合わせ $Q_t^{1:d}$ に依存する．このため，個々の階層の状態のインデックス Q_t^d には実質的な意味は無く，組み合わせ $Q_t^{1:d}$ のみを考えればよい．この特性から，HHMM の状態空間は，1 つのノードが上位の階層の状態の組み合わせ $Q_t^{1:d}$ を表現する木構造で表すことができる．図 3 は，深さ $D=3$，1 階層の状態数 $N=3$ の HHMM の状態空間を表した木構造である．全ての階層の状態の組み合わせを表現するため，木構造の深さは 3 である．また，1 階層ごとに取りうる状態が 3 状態に分かれるため，それぞれのノードは 3 個の子供を持つ．四角で囲んだノードは，互いに状態遷移確率分布 A によって遷移しうる部分 HMM を表す．また，ある状態に関する状態遷移確率および初期状態確率は，その親ノードに依存して一意に決まる．

一般には，それぞれのノードが持つ子供の数が一定でない HHMM や，葉ノードの階層が一定でない HHMM，同じ部分モデル内に葉ノードと内部ノードが含まれているような HHMM を考えることもできる．しかし本稿では簡単のため，それぞれの部分モデルが持つ状態数 N は一定，葉ノードの階層 D も一定とする．つまり，HHMM の状態空間を表す木構造は，完全にバランスした N 分木とする．

4.　活性化 Forward-Backward アルゴリズム

本章では，本研究で提案する HHMM のパラメタ推定アルゴリズムについて述べる．新たに記号を定義し，EM アルゴリズ

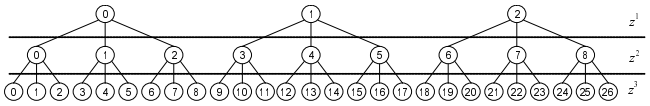

図 4　HHMM の絶対経路状態識別子

ムの枠組みを概説した後，HHMM のパラメタの効率的な再推定式が EM アルゴリズムの枠組みから導出できることを論じる．

4.1　絶対経路状態表現

これまで，状態の識別子は，各階層で独立して 1 から N までの値を持つ Q^d を用いてきた．Q^d は，状態空間の木構造において，相対経路を表す識別子である．一方で，状態遷移確率，初期状態確率，出力確率の分布は全て木構造の絶対経路に依存して与えられる．このため，本稿では，木構造の絶対経路に対する識別番号を状態の識別子とした絶対経路状態識別子 Z^d を定義する．図 4 は，深さ $D=3$，状態数 $N=3$ の絶対経路状態識別子を示したものである．Z^d が与えられると，親ノードは必ず 1 つであるという木構造の特性から，原理的に全ての親ノードを復元することができる．この復元を容易にするため，Z^d の親ノードは必ず $Z^{d-1}=RoundDown(\frac{Z^d}{N})$ となるように値を割り当てる．ここで関数 $RoundDown(\frac{Z}{N})$ は，Z を N で割って，余りを捨てた値を表す．このとき，ある状態 Z^d から親階層の状態 Z^{d-1} を求める関数を，

$$parent(Z^d)=RoundDown(\frac{Z^d}{N})$$

と定義する．反対に，状態 Z^d の子階層の状態が取りうる値の集合を与える関数を，

$$child(Z^d)=\{NZ^d+c : 0 \leqq c < N\}$$

と定義する．また，状態 Z^d から状態遷移確率 A によって遷移できる状態の集合を与える関数を，

$$sib(Z^d)=child(parent(Z^d))$$

と定義する．図 4 は，これらの条件を満たす絶対経路状態識別子を示している．

Z^d は $Q^{1:d}$ に対して明示的な識別子を与えることに対応するが，この識別子を用いることは実装上でも有利である．Q^d を状態の識別子とすると，再帰的なデータ構造を用いて状態とパラメタの管理を実装する必要があるが，Z^d を用いた場合，状態とパラメタは階層ごとに N の d 乗の長さの配列を持てばよい．

絶対経路状態表現に関連した記号の定義を表 2 に示す．初期状態確率 π は，階層 d と絶対経路状態 Z^d を与えることで決まるため，π の空間計算量は $O(N^D)$ である．出力確率 B は葉ノードの状態 Z^D と観測値 O を与えることで決まるため，観測値の種類数を K とすると，空間計算量は $O(N^DK)$ である．状態遷移確率 A は，階層 d と遷移前状態 Z_t^d，遷移後状態 Z_{t+1}^d を与えることで決まるが，遷移可能な状態は $sib(Z_t^d)$ に含まれる N 状態だけなので，A の空間計算量は $O(N^{D+1})$ である．

4.2　活性化前向き・後向き確率

HHMM の状態遷移は，最下層の絶対経路状態 Z^D のマルコフ遷移とみなせる．なぜなら，最下層の絶対経路状態 Z^D は全ての上位階層の状態を復元できるため，全ての階層の状態を完全に表現する識別子になっているからである．このことから，単純には，Z_t^D と Z_{t+1}^D の間の遷移確率 $p(Z_{t+1}^D|Z_t^D)$ をあらかじめ計算することで，HMM と同様の Forward-Backward アルゴリズムを適用することができる．しかし，取りうる最下層状態の数は N^D あるため，遷移の組み合わせは N^{2D} となり，大きな D に関して計算量が問題になる．

表 2　パス状態表現による記号の定義

$Z_t^d \in 1,...,N^d$	時刻 t における階層 d のパス状態
A_{dij}	階層 d における状態 $Z_t^d=i$ のときの状態 $Z_{t+1}^d=j$ への遷移確率
A_{diEnd}	階層 d における状態 $Z^d=i$ のときの終了状態への遷移確率
π_{di}	階層 d における $Z^d=i$ の初期状態確率
B_{ik}	$Z^D=i$ のときの観測値 k の出力確率

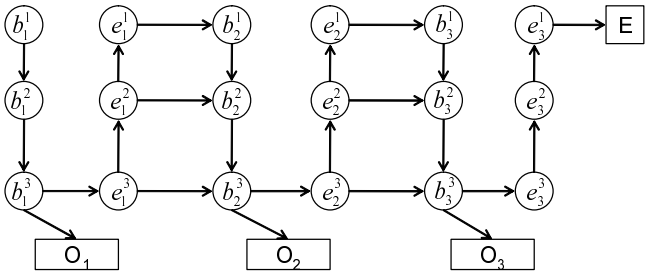

図 5　活性化 Forward-Backward アルゴリズムの確率伝播グラフ

本研究では，全ての Z^D の組み合わせの計算を行わずに，効率的に $p(Z_{t+1}^D|Z_t^D)$ の分布を求める．ここでは，状態遷移を次の 3 種類に分けて考える．すなわち，終了状態への遷移による上方向への垂直遷移，終了状態以外への遷移による水平遷移，そして初期状態の生成による下方向の垂直遷移である．1 回の Z_t^D から Z_{t+1}^D の遷移につき，水平遷移は 1 回だけ起こるが，垂直遷移は最大で $D-1$ 回起こる可能性があり，上方向の垂直遷移と下方向の垂直遷移は同じ回数だけ起こる．垂直遷移の起こる回数は，その時刻に終了状態への遷移が起こる回数である．

HHMM では，ある階層で終了状態に遷移しない場合，それより上位の階層の状態からは水平遷移も垂直遷移も起こらない．ここでは，ある時刻で状態から水平遷移または垂直遷移が起こる事象を，状態の活性化と呼ぶ．上方向の垂直遷移により状態 $Z_t^d=i$ が活性化する確率を $p(e_t^d=i)$，水平遷移または下方向の垂直遷移により状態 $Z_t^d=i$ が活性化する確率を $p(b_t^d=i)$ として，以下のように活性化確率を定義する．

$$p(e_t^d=i) = \begin{cases} p(Z_t^d=i, F_t^{d+1}=1) & (\texttt{if } d<D \texttt{ and } t<T) \\ p(Z_t^d=i) & (\texttt{otherwise}) \end{cases}$$

$$p(b_t^d=i) = \begin{cases} p(Z_t^d=i, F_{t-1}^{d+1}=1) & (\texttt{if } d<D \texttt{ and } t>1) \\ p(Z_t^d=i) & (\texttt{otherwise}) \end{cases}$$

最下層の状態は必ず遷移するため，最下層の状態の活性化確率は，当該状態の滞在確率 $p(Z_t^D)$ と等しい．ある状態の活性化

は，別の活性化した状態からの遷移によってのみ起こるため，初期時刻からの確率伝播によって全ての時刻の活性化確率を求めることができる．図 5 は，活性化確率の依存関係を示した確率伝播グラフである．ここでは観測値系列 $O_{1:T}$ について，活性化状態の前向き確率 α および後向き確率 β を以下のように定義する．

$$\alpha_{e_t^d}(i) = p(e_t^d = i, O_{1:t})$$
$$\alpha_{b_t^d}(i) = p(b_t^d = i, O_{1:t-1})$$
$$\beta_{e_t^d}(i) = p(O_{t+1:T}, F_T^{1:d} = 1 | e_t^d = i)$$
$$\beta_{b_t^d}(i) = p(O_{t:T}, F_T^{1:D} = 1 | b_t^d = i)$$

活性化状態の前向き確率は，図 5 に示すように，$\alpha_{b_1^1} \to \alpha_{b_1^2} \to ... \to \alpha_{b_1^D} \to \alpha_{e_1^D} \to \alpha_{e_1^{D-1}} \to ... \to \alpha_{e_1^1} \to \alpha_{b_2^1} \to \alpha_{b_2^2} \to ... \to \alpha_{b_2^D} \to \alpha_{e_2^D} \to ... \to \alpha_{e_T^1}$ という順で確率伝播を繰り返すことで求めることができる．それぞれの α は，以下のように求められる．

$$\begin{aligned}
\alpha_{b_1^1}(i) &= \pi_{1i} \\
\alpha_{b_1^d}(i) &= \alpha_{b_1^{d-1}}(parent(i))\pi_{di} \ (\texttt{if } d > 1) \\
\alpha_{e_t^D}(i) &= \alpha_{b_t^D}(i) B_{iO_t} \\
\alpha_{e_t^d}(i) &= \sum_{c \in child(i)} \alpha_{e_t^{d+1}}(c) A_{(d+1)cEnd} \ (\texttt{if } d < D) \\
\alpha_{b_t^1}(i) &= \sum_{j \in sib(i)} \alpha_{e_{t-1}^1}(j) A_{1ji} \ (\texttt{if } t > 1) \\
\alpha_{b_t^d}(i) &= \alpha_{b_t^{d-1}}(parent(i))\pi_{di} \\
&\quad + \sum_{j \in sib(i)} \alpha_{e_{t-1}^d}(j) A_{dji} \ (\texttt{if } d{>}1 \texttt{ and } t{>}1)
\end{aligned}$$

後向き確率は，前向き確率と逆の順序で確率伝播を繰り返すことで求められる．すなわち，$\beta_{e_T^1} \to \beta_{e_T^2} \to ... \to \beta_{e_T^D} \to \beta_{b_T^D} \to \beta_{b_T^{D-1}} \to ... \to \beta_{b_T^1} \to \beta_{e_{T-1}^1} \to \beta_{e_{T-1}^2} \to ... \to \beta_{e_{T-1}^D} \to \beta_{b_{T-1}^D} \to ... \to \beta_{b_1^1}$ の順番で計算する．それぞれの β は，以下のように求められる．

$$\begin{aligned}
\beta_{e_T^1}(i) &= A_{1iEnd} \\
\beta_{e_T^d}(i) &= \beta_{e_T^{d-1}}(parent(i)) A_{diEnd} \ (\texttt{if } d > 1) \\
\beta_{b_t^D}(i) &= \beta_{e_t^D}(i) B_{iO_t} \\
\beta_{b_t^d}(i) &= \sum_{c \in child(i)} \beta_{b_t^{d+1}}(c) \pi_{(d+1)c} \ (\texttt{if } d < D) \\
\beta_{e_t^1}(i) &= \sum_{j \in sib(i)} \beta_{b_{t+1}^1}(j) A_{1ij} \ (\texttt{if } t < T) \\
\beta_{e_t^d}(i) &= \beta_{e_t^{d-1}}(parent(i)) A_{diEnd} \\
&\quad + \sum_{j \in sib(i)} \beta_{b_{t+1}^d}(j) A_{dij} \ (\texttt{if } d{>}1 \texttt{ and } t{<}T)
\end{aligned}$$

活性化確率の計算を行うことで最下層の状態の遷移確率 $p(Z_{t+1}^D | Z_t^D)$ が効率的に求められるのは，直観的には以下の理由による．時刻 $t+1$ で状態 Z_{t+1}^D に遷移する確率は，前の時刻の同じ階層の状態 Z_t^D から水平遷移する確率と，上位の階層から下方向垂直遷移する確率の和で与えられる．このとき，ある状態 $Z_{t+1}^D = i$ に水平遷移することができる状態は N 個であり，下方向垂直遷移することができる状態は親状態が唯一であることから 1 個である．このため，親状態の活性化確率さえ分かれば，それぞれの Z_{t+1}^D に対して $N+1$ 通りの遷移確率の和で状態の遷移確率が計算できることになる．最下層の状態は N^D 個あるため，親状態の活性化確率が分かっているとき，$N^D \times (N+1)$ の遷移を考慮すれば全ての最下層状態の確率が求まる．これは，$O(N^{D+1})$ の計算量である．また，親状態の活性化確率も再帰的に $N+1$ 通りの遷移確率の和で計算できるため，計算量は $O(N^{d+1})$ となり，最下層状態の確率を求める計算量よりも小さい．これより，最下層の状態の遷移確率 $p(Z_{t+1}^D | Z_t^D)$ は，活性化確率の確率伝播により $O(N^{D+1})$ の計算量で求めることができる．また，全ての前向き確率および後向き確率は，各時刻での活性化確率の伝播を行うことで求められるため，$O(TN^{D+1})$ の計算量で求めることができる．

これらの活性化前向き・後向き確率を用いることで，HHMM のパラメタ推定は EM アルゴリズムの枠組みで効率良く計算できる．EM アルゴリズムは，潜在変数を含む確率モデルのパラメタ推定を繰り返し計算によって行う手法である．ここでは，まず現在のパラメタを用いて活性化前向き確率および活性化後向き確率を計算し，状態の期待値を求める (E-step)．そして，その期待値を最大にするようにパラメタを更新する (M-step)．E-step と M-step を交互に繰り返すことで，パラメタが学習データの尤度を極大にするように収束することが保証される．

$\sum_{i' \in sib(i)} \bar{\pi}_{di'} = 1$，$\sum_j \bar{A}_{dij} = 1$，$\sum_k \bar{B}_{ik} = 1$ の制約条件の下で期待値を最大にする $\bar{\pi}, \bar{A}, \bar{B}$ は，以下のように求まる．

$$\bar{\pi}_{di} = \frac{g_{\pi di}}{\sum_{i' \in sib(i)} g_{\pi di'}}$$
$$\bar{A}_{dij} = \frac{g_{Adij}}{\sum_{j' \in sib(i), End} g_{Adij'}}$$
$$\bar{B}_{ik} = \frac{g_{Bik}}{\sum_k g_{Bik}}$$

ここで，$g_{\pi di}$，g_{Adij}，g_{Bik} は，それぞれ活性化前向き・後向き確率を用いて以下で与えられる．

$$\begin{aligned}
g_{\pi di} &= \alpha_{b_1^d}(i)\beta_{b_1^d}(i) + \sum_{t=2}^{T} \alpha_{b_t^{d-1}}(parent(d,i))\pi_{di}\beta_{b_t^d}(i) \\
g_{AdiEnd} &= \sum_{t=1}^{T-1} \alpha_{e_t^d}(i) A_{diEnd} \beta_{e_t^{d-1}}(parent(d,i)) \\
&\quad + \alpha_{e_T^d}(i)\beta_{e_T^d}(i) \\
g_{Adij} &= \sum_{t=1}^{T-1} \alpha_{e_t^d}(i) A_{dij} \beta_{b_{t+1}^d}(j) \\
g_{Bik} &= \sum_{t:O_t=k} \alpha_{e_t^D}(i)\beta_{e_t^D}(i)
\end{aligned}$$

これより，活性化前向き・後向き確率を求めることで，HHMM の EM アルゴリズムを実行できる．本アルゴリズムは，活性化前向き・後向きアルゴリズムを求める計算量 $O(TN^{D+1})$ と同じだけの計算量で実行できる．

本アルゴリズムは，状態の活性化確率についての前向き確率

系列長	活性化 Forward-Backward(ms)	一般化 Baum-Welch(ms)
20	109	7348
40	140	40685
60	219	120807
80	296	258899
100	375	504787

表 3　系列長に対する実行時間 (ms)

	D=3	D=4	D=5	D=6
N=3	156(1.93)	250(1.03)	655(0.90)	1983(0.91)
N=4	218(0.85)	687(0.67)	2967(0.72)	10965(0.67)
N=5	345(0.55)	1686(0.54)	8620(0.55)	47435(0.61)
N=6	577(0.45)	3967(0.51)	22372(0.48)	158732(0.57)
N=7	1030(0.43)	7636(0.45)	58441(0.50)	414507(0.50)
N=8	1562(0.38)	13304(0.41)	126349(0.48)	
N=9	2342(0.36)	22411(0.38)	240606(0.45)	

表 4　活性化 Forward-Backward の系列長 100 に対する実行時間 (ms)．括弧内は実行時間を N^{D+1} で割った値

と後向き確率を用いて HHMM のパラメタ推定を行う．一方で，HMM では，全ての状態で必ず遷移が起こることが暗に仮定されているため，活性化確率は状態の滞在確率と等しい．このため，HMM は，状態の滞在確率について Forward-Backward アルゴリズムが適用できる特殊な場合であると解釈できる．本アルゴリズムは，HHMM においても，状態の活性化確率を明示的に求めることで系列長に対して線形な時間計算量で Forward-Backward アルゴリズムを適用できることを示している．

5. 実　　験

提案アルゴリズムを実際に実行し，理論上の計算量でパラメタ推定が行えることを検証する．まず，学習データの系列長に対して計算時間が線形であることを検証するため，系列長を変化させて実行時間の変化を検証する．次に，HHMM の状態数および深さを変化させたときの実行時間の変化を検証する．

ここでは実行時間を得ることが目的であるため，学習データとして人工データを用いる．本実験は，CPU Intel Core2 Duo P8400 2.26GHz，メモリ 2GB の計算機を用いて行った．実装は Java 言語で行い，実行時の最大ヒープメモリサイズを 768MB として実行した．なお，全ての実験で，EM アルゴリズムの繰り返し回数を 50 回とした．

表 3 は，学習データの系列長を 20 から 100 まで変化させて計測した実行時間である．全ての計測結果で，HHMM の状態数は $N = 5$，深さは $D = 3$ を用いた．ここでは，提案する活性化 Forward-Backward アルゴリズムと，従来手法である一般化 Baum-Welch アルゴリズムの実行時間をそれぞれ示す．一般化 Baum-Welch アルゴリズムは，系列長が長くなるに従って，理論上の計算量の通り系列長の 3 乗のオーダーで実行時間が増加している．一方，活性化 Forward-Backward アルゴリズムでは，実行時間は系列長に対して線形に増加しており，一般化 Baum-Welch アルゴリズムと比較して劇的に実行時間が少ないことが分かる．

表 4 は，HHMM の状態数 N，深さ D を変化させたときの，活性化 Forward-Backward アルゴリズムの実行時間である．学習データの系列長は全て 100 である．表中で空欄になっている箇所は，指定した最大ヒープサイズを超えてメモリを必要としたことを意味する．実行時間は，理論上は状態数 N と深さ D に対して $O(N^{D+1})$ であるため，実行時間を N^{D+1} で割った値を表中に括弧付きで示した．実行時間は N と D が大きくなるにつれ増加しているが，括弧内の数値がほぼスケールしていることから，理論上の計算量で実行できているものと考えられる．

6. 結　　論

本研究では，HMM の拡張モデルである階層型隠れマルコフモデルの高速なパラメタ推定アルゴリズムを提案し，時間計算量が従来の手法よりも小さい $O(TN^{D+1})$ であることを示した．

本稿では簡単のため，状態空間が完全にバランスした N 分木で与えられる場合のみ取り扱ったが，木がバランスしていない HHMM や，木構造では表せない下位階層の状態を共有するような HHMM [3] への適用が今後の課題である．

文　　献

[1] D.M. Bikel, R. Schwartz, and R.M. Weischedel. An algorithm that learns what's in a name. *Machine Learning*, 1999.

[2] M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In *Proc. Computer Vision and Pattern Recognition*, 1997.

[3] H.H. Bui. Hierarchical hidden markov models with general state hierarchy. In *Proc. Association for the Advancement of Artificial Intelligence*, 2004.

[4] A. Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.

[5] S. Fine, Y. Singer, and N. Tishby. The hierarchical hidden markov model: Analysis and applications. *Machine Learning*, 1998.

[6] Z. Ghahramani, M.I. Jordan, and P. Smyth. Factorial hidden markov models. *Machine Learning*, 1997.

[7] S. Luhr, H.H. Bui, S. Venkatesh, and G.A.W. West. Recognition of human activity through hierarchical stochastic learning. In *Proc. Pervasive Computing and Communication*, 2003.

[8] K.P. Murphy and M.A. Paskin. Linear time inference in hierarchical hmms. In *Proc. Neural Information Processing Systems*, 2001.

[9] N.T. Nguyen, D.Q. Phung, and S. Venkatesh. Learning and detecting activities from movement trajectories using the hierarchical hidden markov models. In *Proc. Computer Vision and Pattern Recognition*, 2005.

[10] M. Skounakis, M. Craven, and S. Ray. Hierarchical hidden markov models for information extraction. In *Proc. International Joint Conference on Artificial Intelligence*, 2003.

[11] C. Vogler and D. Metaxas. Parallel hidden markov models for american sign language recognition. In *Proc. International Conference on Computer Vision*, 1999.

[12] M. Weiland, A. Smaill, and P. Nelson. Learning musical pitch structures with hierarchical hidden markov models. In *Proc. Journees Informatiques Musicales*, 2005.

[13] L. Xie, S. Chang, A. Divakaran, and H. Sun. Learning hierarchical hidden markov models for video structure discovery. Technical report, 2002.

[14] S. Yu. Hidden semi-markov models. *Artificial Intelligence*, 2010.