Vol.2010-ARC-187 No.11
Vol.2010-EMB-15 No.11
2010/1/29

TLDP 法のパイプライン化による省メモリな連続単語音声認識回路

島崎 亮[†1] 中村一博[†1]
高木一義[†1] 高木直史[†1]

本稿では、TLDP(Two Level Dynamic Programming) 法のパイプライン化による省メモリな連続単語音声認識回路の構成法を提案する。入力音声内の単語列を認識する連続単語音声認識では、単語の位置と種類を特定する為に音声内の各位置において繰り返し孤立単語音声認識処理が行われる。この処理は HMM 出力確率計算が主であり、これが膨大な計算時間やメモリ量を必要とする。提案する構成法では TLDP 法における孤立単語音声認識処理をグループ化し結果を段階的に出力する事で、その結果を基に最適な単語列を求める処理で必要なメモリ量の削減を図っている。

Memory Efficient Pipelined TLDP Architecture for Connected Word Speech Recogntion

Ryo Shimazaki, Kazuhiro Nakamura,
Kazuyoshi Takagi and Naofumi Takagi

In this manuscript a memory efficent piplined TLDP architecture for the connected word speech recognition is presented. The connected word speech recognition identifies a sequence of words from input speech. First, it iteratively processes multiple isolated word speech recognition to identify word's positions and word. After that, the best word sequence is computated from the first level result. The most time consuming part of the isolated word speech recognition process using HMM is the process to calcuate the output probability. The computaion requires a large mount of memory and processing time. The proposing approach tries to reduce memory size which is needed at last half processes, by counting off multiple isolated word speech recognition processes into some groups and outputing this results in a stepwise fashion.

1. はじめに

携帯電話や PDA の普及に伴い、音声や映像の認識によるより高度なヒューマンインターフェースが求められている。その一つとして音声認識が注目され、音声認識機能を持ったモバイル機器の研究・開発が行われている。音声認識システムをソフトウェアで実現する場合、計算量の増加によるシステムのパフォーマンスの低下やプロセッサの消費電力の増加が考えられる[1)]。必要とされるデバイスの特性上、組込用プロセッサを使用することが予想されるが、この場合は膨大な計算量に対するプロセッサの処理速度の問題が浮上する。ヒューマンインターフェースとして使用される事を前提としたこのような機能はリアルタイム性が求められ、よって組込用プロセッサもシステムに適しておらず、結論として専用ハードウェアで実現されることが望ましいと言える。しかしハードウェアでの実現においても、認識処理の為の膨大な計算量や、語彙数の増加による学習データ記憶の為のメモリ容量の増加が課題として挙がっている。

音声内の複数単語からなる単語列を認識する連続単語音声認識処理[2)] では、音声内に含まれる各単語の位置を特定する処理を行う。孤立単語音声認識処理[3)] は HMM を用いた出力確率計算が主であり、これが膨大な計算時間やメモリ量を必要とする要因になっている。

音声認識回路には連続音声認識や単語音声認識など音声認識の方法により様々な回路の構成方法がある。その中でも孤立単語音声認識回路と連続単語音声認識回路には類似点が多い。主要な連続単語音声認識の処理手順は、処理の前半部に孤立単語音声認識を繰り返し実行する。この部分に本処理の計算上の負荷が集中している。

本研究では、連続単語音声認識を行う TLDP(Two Level Dynamic Programming) 法と呼ばれる手法をパイプライン化することにより、省メモリな連続単語音声認識回路の構成法を提案する。音声内の単語列を認識する連続単語音声認識では、各単語の位置を特定する為に音声内の各位置において孤立単語音声認識処理を行う。これは連続単語音声認識を行う TLDP 法の第一段階処理に含まれる。第一段階処理が終了したらこの結果を利用して第二段階処理に移行する。本研究では TLDP 法の第一段階処理である孤立単語音声認識処理をグループ化し結果を段階的に出力する事で、第二段階処理に必要なメモリ量の削減を図っている。

†1 名古屋大学
Nagoya University

Vol.2010-ARC-187 No.11
Vol.2010-EMB-15 No.11
2010/1/29

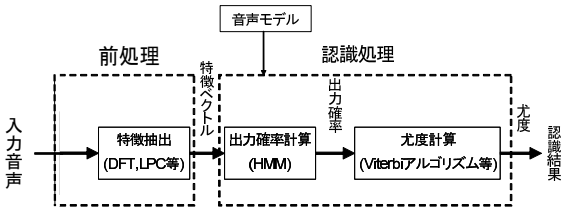

図 1 HMM を利用した音声認識の流れ
Fig. 1 flow of voice recognition with HMM.

2. 音 声 認 識

音声認識は人が発した音声を入力とし、予め登録された音声モデルから入力信号に最も近い音声モデルを出力する処理である。音声モデルは、単語や音素、音韻等の認識対象をモデル化したものであり、音声認識などには時間的な変動に頑健な隠れマルコフモデル（Hidden Markov Model; HMM）が使用される[2)3)]。孤立単語音声認識では、あらかじめ単語毎に対応するモデルを用意しておき、入力音声とこれを比較して最も近いモデルを出力する。

単語音声認識には孤立単語音声認識の他に連続単語音声認識がある。孤立単語音声認識は認識対象の音声が一つの単語 (HMM) であるという仮定に基づいているのに対し、連続単語音声認識は認識対象が複数の単語（単語列）から構成されているという前提に基づいている。

2.1 孤立単語音声認識

HMM を利用した孤立単語音声音声認識の処理は図 1 に示すように大きく 3 つにわかれる。特徴抽出、出力確率計算、Viterbi アルゴリズム等による尤度計算である。特徴抽出処理は認識処理の前処理として、音声信号から認識処理に必要な情報を特徴ベクトルとして取り出す。これは通常 DSP などによりリアルタイムに処理される[4)]。次に特徴ベクトルを入力として、出力確率計算を行う。最後に Viterbi アルゴリズムによる尤度計算を行う。

以下に出力確率の計算式を記す。出力確率計算は加算、減算、乗算、累算により、式 (1) の $\log b_j(\mathbf{o}_t)$ を求める計算である。

$$\log b_j(\mathbf{o}_t) = \omega_j + \sum_{s=1}^{P} \sigma_{js}(o_{ts} - \mu_{js}) \quad (i = 0, 1, 2, \ldots, N) \tag{1}$$

ここで $\mathbf{o}_t$ は時刻 t における入力音声の特徴ベクトル $\mathbf{o}_t = (o_{t0}, o_{t1}, \ldots, o_{tP})$、$P$ は次元数である。一つの HMM モデルは N 個の状態を持っており。HMM 毎にある状態 i からある状態 j へと移動する確率 a_{ij} が決まっている。HMM の出力確率 $b_j(\mathbf{o}_t)$ は状態 j において、指定したシンボル $\mathbf{o}_t$ が出力される確率を表している。μ, ω, σ の各ベクトルはモデルによって値が異なり、出力確率計算を行う前にあらかじめ求めておくことが可能な各 HMM の学習パラメータである。

次に、このようにして求まった出力確率から Viterbi アルゴリズムを用いて、モデルにおける特徴ベクトル系列の尤度を求める。ここで尤度とはそのモデルにおいて特徴ベクトル系列が出力される尤もらしさである。尤度は次の式 (2),(3),(4) によって計算される[3)4)]。

$$\delta_0(j) = \log \pi \qquad (1 \le j \le N) \tag{2}$$

$$\delta_t(j) = \max_{i=j-1,j} [\delta_{t-1}(i) + \log a_{ij}] + \log b_j(\mathbf{o}_t) \qquad (1 \le t \le T,\ 1 \le j \le N) \tag{3}$$

$$P^* = \max_{1 \le j \le N} [\delta_T(j)] \qquad (t = T) \tag{4}$$

T は特徴ベクトル系列の総数を表す。$\log \pi$ は HMM 毎に決められた学習パラメータの一つである。この尤度 P^* を求める計算を最尤推定という。あらかじめ学習されている単語 HMM の数だけこの出力確率計算と最尤推定を繰り返し、一番尤もらしいモデルが認識結果として出力される。

2.2 TLDP 法による連続単語音声認識

連続単語音声認識を行う手法の一つに、TLDP(Two-Level Dynamic Programming) 法がある。TLDP 法には大きく二つの処理段階があり、孤立単語音声認識の処理を繰り返す第一段階と、その結果を基に最適な単語列を求める第二段階に分けられる。第一段階の処理では入力音声の特徴ベクトル系列に対し、複数の部分音声区間で孤立単語音声認識を行う。第二段階の処理では第一段階で求めた各区間の尤度を基に、入力音声全体にわたり最適になるような各部分音声区間の結果を足し合わせた尤度を求める。

第一段階処理では、総区間数が M の対象音声内に対し、開始地点を $s(1 \le s \le M)$、終了地点を $e(s < e \le M)$ とする s から e までの各区間を部分音声 (s, e) とし、この部分音声について孤立単語音声認識処理を行う (図 2)。そして式 (5) に示す $\tilde{P}(s, e)$ を算出する。

第二段階処理では以下の式 (6),(7) の計算により単語列の尤度計算を行う。

Vol.2010-ARC-187 No.11
Vol.2010-EMB-15 No.11
2010/1/29

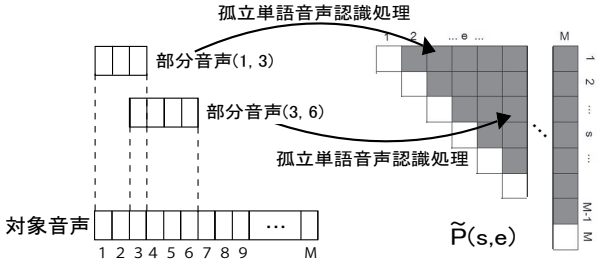

図 2 部分音声 (s, e) とその孤立単語音声認識処理結果 $\tilde{P}(s, e)$
Fig. 2 Representation of speech segment(s,e) and $\tilde{P}(s, e)$.

$$\tilde{P}(s,e) = \max_{1 \le v \le V} [P^*(v,s,e)] \tag{5}$$

$$D_l(e) = \max_{1 \le s < e} [\tilde{P}(s,e) + D_{l-1}(s-1)] \quad (1 < l \le M/2) \tag{6}$$

$$D_1(e) = \max_{1 \le s < e} [\tilde{P}(s,e)] \tag{7}$$

$\tilde{P}^*(v,s,e)$ は部分音声 (s,e) の単語 v に対する尤度計算の結果である。$\tilde{P}(s,e)$ はその区間におけるベストマッチな単語の尤度、すなわち全単語内で最も高い尤度である。V は認識候補の単語の総数である。$D_l(e)$ は位置 e で終わる l 個の単語から成る単語列の尤度であり、式 (6) より $D_l(e)$ は、$l-1$ 個の単語からなる、$s-1$ を終端とする部分音声についての尤度 $D_{l-1}(s-1)$ と、位置 s から e までの部分音声におけるベストマッチな単語の尤度 $\tilde{P}(s,e)$ の二つの尤度の和から求められる。全ての s に対しこれを求め、最大の尤度になる単語列が選ばれる。これを終端位置 $e = M$ まで求める事で、音声全体において最も確からしい単語列が求められる。最大の $D_l(M)$ に対応する単語からバックトラッキングを行うことで連続単語音声認識処理が完了する。

TLDP 法を実現する回路の一例として、Kim らの連続単語音声認識回路の第二段階処理部を図 3 に示す。Kim らの手法では、第二段階処理回路部にシストリックアルゴリズムが用いられている。M 個の PE(Processing Element) と $L \times M$ 個のレジスタが配列状に配置されており、そのハードウェア構成の中でデータが単純な規則に基づいて循環、処理されることで式 (6),(7) の計算がなされる。L は認識する単語列中に存在する可能性のある最大の単語数を表し、この場合 $M/2$ である。図 4 に示す PE は加算器、比較器、図において z^{-1} で示されている b ビットレジスタから構成され、M 列に配置され並列に動作する。図 5 に

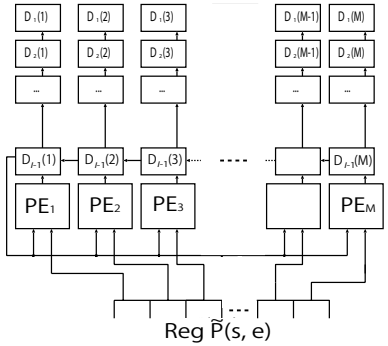

図 3 Kim らの第二段階処理部の回路構成[2)]
Fig. 3 Conventional architecture for second level processing.

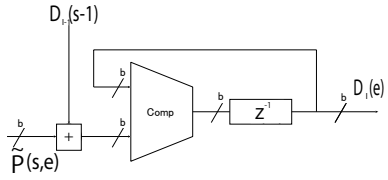

図 4 PE 内部の回路構成
Fig. 4 architecture of PE.

この回路の動作を示す。PE によって求められた第二段階処理の結果 $D_l(e)$ は各 PE 上部にある記憶領域に単語数の少ない結果の順に格納される。(図 5：20 行目)

3. 提案する連続単語音声認識回路

本節では 2 節で述べた従来の回路構成よりも効率的な回路構成を提案する。TLDP 法では第一段階における複数の部分音声に対する孤立単語音声認識処理に計算上の負荷が集中する。また、第一段階の処理結果を全て格納 (図 2 $\tilde{P}(s,e)$, 図 3 $Reg\tilde{P}(s,e)$) しなければならないので、大量の回路内メモリを必要とする。そこで、第一段階と第二段階における処理のパイプライン化手法とそれに適した回路構成法を提案する。具体的には第一段階における孤立単語音声認識処理をグループ化し、この結果を段階的に出力させる。これにより第二段階処理に必要なメモリ量を削減出来るような第二段階処理回路を構成する事を可能にする。

3.1 孤立単語音声認識処理のグループ化による第一段階処理の高速化

第一段階処理では孤立単語音声認識処理が繰り返し行われる。2 節で述べた手法ではこの複数の孤立単語音声認識処理が全て終了しなければ第二段階処理に進む事は出来なかった。そこで我々はこれらの処理をいくつかのグループに分割し、グループ毎に段階的な第一段階処理と第二段階処理を行う。本手法により、第一段階処理はその計算結果を局所的に早く出力できるようになる。このグループ化を行う事により、図 2 に示した連続する出力確率計算と尤度計算にかかる処理時間を短縮できる。例えば部分音声 $(s,e) = (1,4)$ について計算し

Vol.2010-ARC-187 No.11
Vol.2010-EMB-15 No.11
2010/1/29

入力： 第一段階処理結果 $\tilde{P}(s,e)$ $(1 \le s < e \le M)$
出力： 第二段階処理結果 $D_l(e)$ $(1 \le l \le M/2,\ e = 1,2,\ldots,M)$

1: 各レジスタ $D_{l-1}(e)$ の値を 0 に初期化 $(e = 1,2,\ldots,M)$
2: for $l = 1$ to $M/2$
3: /* ここから各 PE_e が並列に処理を行う $(e = 1,2,\ldots,M)$*/
4: for $s = 1$ to M
5: if $s+1 < e$ then
6: if $l = 1$ then
7: if $z^{-1} < \tilde{P}(s,e)$ then $z^{-1} = \tilde{P}(s,e)$ endif
8: /* 各 PE_e が単語数 1 の場合の処理を行う*/
9: endif
10: if $s > 2 \wedge e > 2 \times l$ then
11: if $z^{-1} < \tilde{P}(s+1,e) + D_{l-1}(s)$ then $z^{-1} = \tilde{P}(s+1,e) + D_{l-1}(s)$ endif
12: /* 各 PE_e の z^{-1} の値の更新*/
13: endif
14: endif
15: if $s > 2$ then
16: 各レジスタ $D_{l-1}(e)$ の値を $D_{l-1}(e-1)$ にシフト
17: endif
18: endfor
19: PE_e の z^{-1} の値を $D_{l-1}(e)$ にコピー
20: レジスタ D_{l-1}(e) の値を出力格納部にコピー
21: /* ここまで各 PE_e が並列に処理を行う $(e = 1,2,\ldots,M)$*/
22: endfor

図 5 Kim らの第二段階処理のハードウェアアルゴリズム
Fig. 5 Kim's hardware algorithm

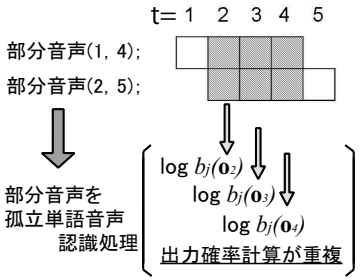

図 6 複数の部分音声に対する孤立単語音声認識処理における出力確率計算の重複
Fig. 6 Computations with speech segment (s,e)=(1,4)→(2,5).

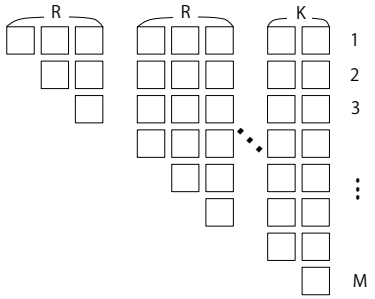

図 7 部分音声のグループ化
Fig. 7 Group of speech segment.

た後に部分音声 $(2,5)$ について計算する場合、図 6 のように区間 $(2,4)$ における $\log b_j(\mathbf{o}_t)$ の計算は重複する。$(1 \le j \le N)$,$(2 \le t \le 4)$ この性質を活用すると、部分音声 (s,e) についての認識処理を行いながら部分音声 $(s+1,e+1)$ についての計算を行えば、部分音声 (s,e) についての計算の過程で算出された $\log b_j(\mathbf{o}_t)$ が再度使用される事になる。この計算の重複を活用しつつ部分音声 (s,e) における尤度 $\tilde{P}(s,e)$ を求める事で、処理の高速化が図れる。計算の手段として、先に示した部分音声を複数のグループに分ける処理を利用する。これは音声の総区間 $(1,M)$ を R 区間ずつ M/R のグループに分け、この分割した各グループについて R 並列に最尤推定を行う事で実現できる。この時 $K = M \bmod R$ とする (図 7)。

3.2 第一段階処理と第二段階処理のパイプライン化による省メモリ化

Kim らによる TLDP 法の回路構成では、第一段階処理結果を格納する為の記憶領域として $b \times M^2/2$ ビット分を用意する必要があった。これは総区間数 M の部分音声において、存在する全ての部分音声 (s,e) に対する孤立単語音声認識処理の結果を格納しなければならない為である。これに対し提案手法では、第一段階処理結果を格納する領域は $b \times MR$ ビット分で足りる。孤立単語音声認識処理のグループ化により、第一段階処理結果 $\tilde{P}(s,e)$ もグループ化され、部分音声 (s,e) において位置 e の早い順に、グループ毎に出力されるようになる。これにより一度に出力される $\tilde{P}(s,e)$ が小さくなる為である。提案手法ではこの小さい第一段階処理結果の格納領域に対応して第二段階処理を行う。回路の構成を図 8 に示す。

PE はグループ幅数の R だけ配置される。この PE に入力される中間結果 $D_{l-1}(e)$ を格納するメモリの要素数は $2R$ となる。これは直前に求められた $D_{l-1}(e)$ と前グループで求められた $D'_{l-1}(e)$ とを区別して保持するためである。$D_l(e)$ を求める際、$D_l(e_1)$ が既に求められていれば $\tilde{P}(s,e_1)$ $(e_1 < e)$ は直接必要としない。つまり、計算後必要のなくなった $\tilde{P}$ のメモリ領域に次のグループの $\tilde{P}$ を格納する事で、第二段階におけるメモリの節約が図れる。提案する回路のアルゴリズムでは、これまで用いられてきたシストリックアルゴリズムに以上のようなパイプライン処理を組み合わせる。

グループの幅を R とした場合の提案手法による第二段階の具体的な処理の流れを、提案する回路構成におけるハードウェアアルゴリズムとして図 9 に示す。孤立単語音声認識処理のグループ化により、第一段階処理の結果は e の小さい順に R 個ずつ出力される。この為第二段階処理回路には入力として $\tilde{P}(s,e|1 \le s \le R, s < e \le R+1)$ が与えられることになり、従来手法と同様に、単語数 l の少ない順に第二段階処理結果を求められる。まず $D_1(2),\ldots,D_1(R+1)$ が求まる (図 9：8 行目)。次にこの計算結果と第一段階処理結果を使用して、$D_2(e)$ を求める。単語数が 2 の場合、一単語を認識可能な部分音声の長さの制限か

Vol.2010-ARC-187 No.11
Vol.2010-EMB-15 No.11
2010/1/29

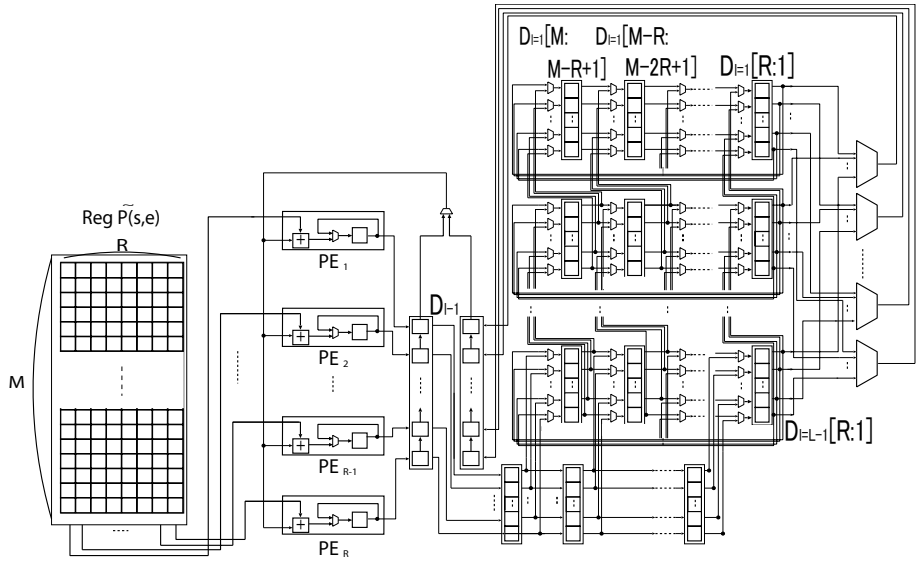

図 8 提案する第二段階処理部の回路構成
Fig. 8 Pipelined TLDP architecture for second level processing.

ら、求められる $D_2(e)$ は $4 \leq e \leq R+1$ となる。一般化すると、単語数 l において、求める事が可能な第二段階処理結果 $D_l(e)$ の終端位置 e は、$2l \leq e$ となる。最初の入力グループ $\tilde{P}$ が求められる最大の単語数まで第二段階処理が進んだら、この第二段階処理結果を格納し、次の入力グループが与えられるまで待機する (図 9：3 行目)。

ここで初めに与えられた $\tilde{P}$ に代わり、第一段階の次グループの認識結果 $\tilde{P}(s,e|1 \leq s \leq 2R, s < e \leq 2R+1)$ が読み込まれる。新たな $\tilde{P}$ が与えられると、まず $D_1(R+2), \ldots, D_1(2R+1)$ が求まる (図 9：8 行目)。次に、第一段階の処理結果と得られた第二段階の処理結果を使用して、単語数 2 の場合の第二段階処理結果 $D_2(e)$ を求める。この際すでに格納した前グループでの第二段階処理結果、つまり $D_1(2), \ldots, D_1(R+1)$ が処理に必要になる。この為 R サイクル毎に D_{l-1} 内に前グループでの結果を読み込ませる (図 9：13 行目)。これを入力として各 PE が $D_2(e)$ を求める ($R+2 \leq e \leq 2R+1$)。このように、直前で計算された第二段階処理結果と、前のグループで計算された第二段階処理結果を利用して、単語数の大きな第二段階処理結果を求める (図 9：19,22 行目)。入力されたグループが計算可能な最大の単語数の結果まで求められたら、次のグループの認識結果を読み込ませ、同様に単語数の低い第二段階処理結果から求めていく。このような繰り返しを第一段階処理の結果の最終グループに達するまで行い、これより最終的なスコア $D_l(M)$ が求められる。

4. 詳細設計と評価

提案手法に基づく連続単語音声認識回路の第二段階処理部を Verilog-HDL で記述し評価を行った。設計した回路の構成と諸元は以下の通りである。

本回路は大まかに 3 つの領域に分けられる。$\tilde{P}(s,e)$ を格納する入力データ格納領域 (図 8 における左側部分：$Reg\tilde{P}(s,e)$)、D_{l-1} の値を保持し D_l を計算する演算領域 (図 8 における中央部分：R 個の PE, $2R$ 個のレジスタ D_{l-1})、計算された D_l の値を格納する結果格納領域 (図 8 における右側部分：R レジスタからなるレジスタ集合 D_l) である。大幅にメモリ量を削減出来るのは演算領域と入力データ格納領域の二つである。従来の手法では PE を部分音声の分割数だけ用意しなければならないが、提案手法では PE をグループの幅 R 分だけ用意すればよいので、PE 数だけ $\tilde{P}(s,e)$ を格納する為のメモリ量が削減される。

構成される PE やデータ格納領域の個々のパラメータを同一にし、これを従来手法と提案手法それぞれのアーキテクチャに従って第二段階処理回路を構成した。設計には ROHM0.18um スタンダードセルライブラリを利用し、Design Compiler で動作合成を行った。部分音声の総区間数は $M = 40$、提案手法におけるグループの大きさは $R = 6$、認識結果のデータサイズは $b = 24bit$ と $12bit$ のそれぞれの場合について設計した。

表 1 に設計した入力データ格納領域における理論的なメモリ量と、論理合成された回路の面積値と最大遅延を示す。表より、データのビット幅に関わらず、提案手法に基づく回路の方がメモリ量の削減により従来と比較して少ない面積で実現されていることが分かる。

表 2 に演算領域の面積と最大遅延を示す。本領域は PE 内部に尤度を格納する記憶領域を持つので PE 数が M から R 個に削減されることで提案手法に基づく回路の方が小面積化

表 1 入力データ格納領域の設計
Table 1 Area and dely of data memory area.

	従来手法 (24bit)	提案手法 (24bit)	従来手法 (12bit)	提案手法 (12bit)
必要なメモリ量 [bits]	$24 \times M^2/2$	$24 \times MR$	$12 \times M^2/2$	$12 \times MR$
	$M = 40, R = 6$			
面積値	5,822,842	873,501	2,969,566	445,609
遅延	2.80	2.80	2.84	2.87

Vol.2010-ARC-187 No.11
Vol.2010-EMB-15 No.11
2010/1/29

入力： 第一段階処理結果の連続 $\tilde{P}_r(s,e)$ $(r=1,\ldots,M/R)$
出力： 第二段階処理結果 $D_l(e)$ $(1 \le l \le M/2,\ e=1,2,\ldots,M)$

1: レジスタ $D_{l-1}(e)$ の各値を 0 に初期化 $(e=1,2,\ldots,R)$
2: for $r=1$ to M/R
3: 第 r グループの第一段階処理結果 $\tilde{P}_r(s,e)$ が入力されるまで待機
4: for $l=1$ to $r \times R/2$
5: /* ここから各 PE_e が処理を並列に行う $(e=1,2,\ldots,R)$*/
6: if $l=1$ then
7: for $s=1$ to $M+1$
8: if $s < r \times R + e$ then $z^{-1} = \max_{s,e}[\tilde{P}_r(s,e)]$ endif
9: /* 各 PE_e が単語数が 1 の時の処理を行う*/
10: endfor
11: else
12: for $s=2$ to $M+1$
13: if $s-2 \bmod R = 0 \wedge s \le M-R$ then
14: $D'_{l-1}(e) \leftarrow$ 前グループの第二段階処理結果 $D_{l-1}(s)$
15: endif
16: if $((r-1) \times R + e \ge 2l) \wedge (2(l-1) \le s < (r-1) \times R + e)$ then
17: /* 以下で各 PE_e の z^{-1} の値の更新*/
18: if $s \le M-R$ then
19: $z^{-1} = \max[D_{l-1}(e) + \tilde{P}_r(s+1,e)]$
20: /* 各 PE_e は直前に計算された第二段階処理結果を使用*/
21: else
22: $z^{-1} = \max[D'_{l-1}(e) + \tilde{P}_r(s+1,e)]$
23: /* 各 PE_e は前グループで計算された第二段階処理結果を使用*/
24: endif
25: endif
26: レジスタ $D_{l-1}(e)$ の値を $D_{l-1}(e-1)$ にシフト
27: endfor
28: endif
29: $D_{l-1}(e)$ の値をスタック
30: レジスタ $z^{-1}(e)$ の値を $D_{l-1}(e)$ にコピー
31: /* ここまで各 PE_e が処理を並列に行う $(e=1,2,\ldots,R)$*/
32: endfor
33: スタックされた $D_{l-1}(e)$ の値を全て第二段階処理結果格納部にコピー
34: 第二段階処理結果を巡回整理
35: endfor

図 9 提案する第二段階処理のハードウェアアルゴリズム
Fig. 9 Hardware algorithm for second level processing of pipelined TLDP.

されている。また同様に、直前の処理結果 D_{l-1} を格納する為の記憶領域も削減されている事も分かる。

5. ま と め

本研究では TLDP 法のパイプライン化による省メモリな連続単語音声認識回路の構成法を提案した。これに基づいて設計した第二段階処理回路における入力データ格納部、演算処理部において、大幅なメモリ量の削減と小面積化を実現できた。

謝辞 本研究の一部は、東京大学大規模集積システム設計教育センターを通じ、日本シノプシス株式会社、ローム株式会社の協力で行われたものである。

参 考 文 献

1) A. Lee, Kawahara, T. and Shikano, K.: Julius . an opensource real-time large vocabulary recognition engine, Proc. European Conference on Speech Communication and Technology, pp. 1691-1694 (2001).
2) Y, Kim. and H, Jeong.: A Systoric FPGA Architecture of Two-Level Dynamic Programming for Connected Speech Recognition, IEICE TRANS.INF&SYST., VOL.E90-D, NO.2 (2007).
3) Yoshizawa, S. Wada, N. and Hayanaga, N.: Scalable Architecture for Word HMM-based Speech Recognition and VLSI Implementation in CompleteSystem, IEEE TRANS. ON CIRCUITS AND SYSTEMS, Vol. 53, No.1, pp. 70-77 (2006).
4) Miura, K. Noguchi, H. Kawaguchi, H. and Yoshimoto, M.: A Low Memry Bandwidth Gaussian Mixture Model(GMM) Processor for 20,000-word Real-Time Speech Recognition FPGA System, Proc. of the International Conference on Field-Programmable Technology, pp. 341-344 (2008).
5) Hashimoto, J. Eguchi, A. Saituji, M. Yamada, A. and Kanbe, T.: An Output Probability Computation Circuit Design for Real Time Speech Recognition, SASIMI(2007)

表 2 演算領域の設計
Table 2 Area and delay of computing area.

	従来手法 (24bit)	提案手法 (24bit)	従来手法 (12bit)	提案手法 (12bit)
必要なメモリ量 [bits]	$24 \times 2M$	$24 \times 3R$	$12 \times 2M$	$12 \times 3R$
	$M=40, R=6$			
面積値	360,583	49,811	167,003	25,012
遅延	8.16	8.35	5.54	5.56