TR-IT-0067

フレーム同期型 SSS-LR による文音声認識における効率的な探索方式の検討

A study on efficient search method for sentence recognition using Frame Synchronous SSS-LR

門前 聖康
Monzen Seikou

清水 徹
Tohru Shimizu

松永 昭一
Matsunaga Shoichi

1994.8.31

前回の研究では、拡張 LR 構文解析法を用いた連続音声認識手法として、フレーム同期型 SSS-LR について検討し、報告した (TR-IT-0051)。文節音声認識実験結果から、フレーム同期型 SSS-LR の有用性を報告した。また、小規模ながら文音声認識実験を行ない、その実現可能性を示した。そこで、本稿では、このフレーム同期型 SSS-LR を用いた文音声認識について検討した結果を報告する。

文音声認識は文節認識に比べ探索空間が膨大で、また、文節間での無音や冗長語の出現により認識精度が大きく低下するため、認識精度の高いモデルの構築と効率的な探索手法とが求められる。本研究では、後者の要求に答えるために、より効率的な探索方式について検討した。

目次

第 1 章

フレーム同期型 SSS-LR による文連続音声認識

前回のテクニカルレポート (TR-IT-0051) で報告したフレーム同期型 SSS-LR について、再度概説する。

1　フレーム同期型 HMM-LR 認識系

フレーム同期型 HMM-LR は、HMM を用いて One-Pass Viterbi サーチに基づく音響照合を行なう探索部と、拡張 LR 構文解析法を用いて統語解析を行なう統語解析部とから構成される。フレーム同期型 HMM-LR の認識系の構成図を図 1.1に示す。この時の音響モデルとして HMnet[3] を、統語解析法として文脈依存型 LR パーザ [1] を用いる場合がフレーム同期型 SSS-LR である。

1.1　探索部

探索部では、One-Pass Viterbi サーチに基づき、入力音声の各フレームにおいて音素照合を行なうと同時に、音素や単語等の連鎖についての統語的制約を逐一適用していく。入力音声の最終フレームにおける文法仮説中の、最も尤度の高い仮説を認識結果として出力する。One-Pass サーチでは、言語知識から得られる制約を、音素照合に対して動的に適用する事ができるため、全体の探索空間を大幅に削減できる。従ってラティスパージング等の手法に比べてオーバーヘッドが少なく、認識精度や処理量の点から、有効なサーチ方法である。

1.2　統語解析部

統語解析部では、LR テーブルと文脈自由文法を用いて、音素予測を行なう。走査した各音素に対してスタックを更新し、更新されたスタックにおける予測音素を計算する。

状態ネットワーク

フレーム同期処理では、入力音声のフレーム毎に構文解析部を駆動する可能性があるため、様々な状況のスタックを同時に表現しなければならない。そこで、LR 解析におけるスタック構造をグラフ構造に展開し、スタック表現の効率化を図る。以下このグラフ構造化スタックを状態ネットワークと呼ぶ。状態ネットワークの例を図 1.2に示す。状態ネットワークでは、スタック中の各状態をネットワーク上のノードで表わす。LR 解析の初期状態は、ネットワークのルートノードに対応する。スタックの構造はネットワークノード系列で表現される。即ち、ルートノードから各ネットワークノードへ至るノードの系列が、そのノードをスタックの最上位 (stack top) とするスタックを表現している。各ノード間を結ぶリンクは、リンク元のノー

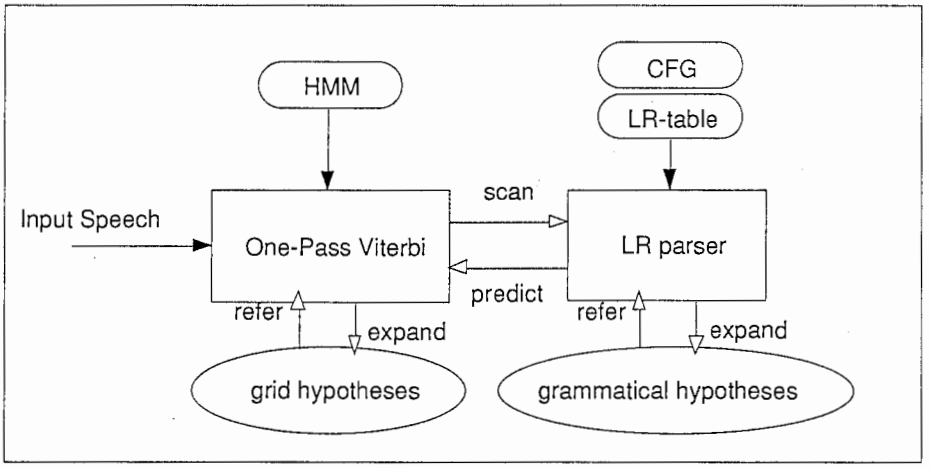

図 1.1: 認識系の構成図

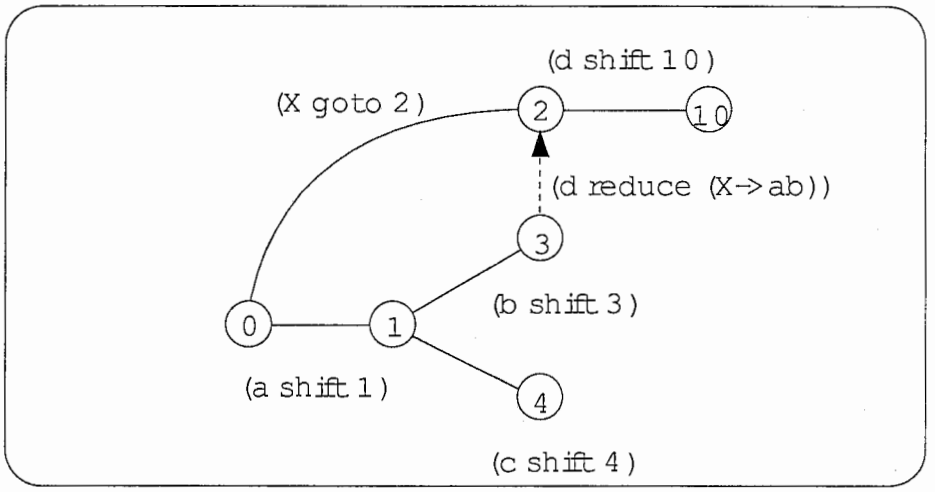

図 1.2: LR 状態ネットワーク

ドから、アクションを実行した際の遷移を意味し、スタックに積まれた状態の上下関係を表わす。

先読み処理

統語解析部は、LR 解析によって音素予測を行なうモジュールであるが、文脈依存型 HMM である HMnet を用いる場合、当該 (予測) 音素だけでなく、先行した音素、後続可能な音素を考慮する必要がある。特に後続音素に関しては、各予測音素について、さらに後続可能な音素まで予め計算しなければならない。

そこで、構文解析において状態ネットワークを展開する際に、各々のスタックでの予測音素だけでなく、各々の予測音素を疑似的に走査したものとみなし、さらに構文解析を実行する。その時に予測されている音素を、現在の後続音素とする。但し、この先読み処理では、単に、可能な動作をトレースするだけで、状態ネットワークは展開しないので、認識に不要なネットワーク、即ち探索されない部分を構築することはない。

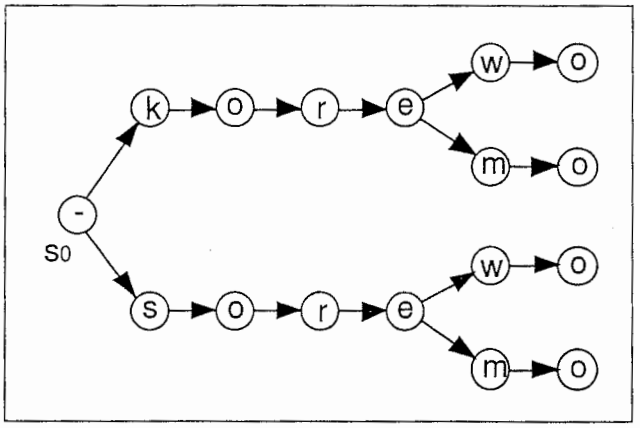

図 1.3: 音素履歴木の例

1.3　探索部と統語解析部の分離

音声認識では、認識の最終段階には One-Pass Viterbi により最適な経路を決定できるものの、探索途中においては一意に音素系列を決定できないため、同時に複数の文法仮説を保持しておく必要がある。しかし、生成可能な文法仮説の総数は組み合わせ的に爆発するため、全ての仮説を生成することは不可能である。

また、探索部における音素照合では音素系列が一意化されれば各音素系列に対する尤度も決定するが、一方統語解析部では、文法上の曖昧さから、一つの音素系列に対して、複数の文法仮説を作成し得る。音素系列が同じであれば、それらの文法仮説の音響尤度は全て等しいが、これらを別々の仮説として扱うと、同じ音響照合を何度も駆動することになり極めて冗長である。

従って音素照合における曖昧さと統語解析部における曖昧さを、各々に反映させないよう、分離する必要がある。

音素照合の曖昧さが、できるだけ統語解析部の負荷とならないために、動的に文法仮説を展開する。音素照合部での音素検出に際してのみ統語解析を駆動させ、探索の対象となりうる文法仮説のみを展開することで、生成する仮説数を極力小さく抑える。

また、統語解析部の曖昧さが、音素照合部の無駄を生じさせないために、音素系列が同じ文法仮説をパッキングする [4]。音素系列を表現する音素履歴木を構築する段階で、音素系列が同じ文法仮説を一まとめにする。探索部へは、展開可能な音素履歴、即ち後続可能な予測音素という情報だけを与える。

音素履歴木は、音素を木のノードに対応させた木構造のデータである。図 1.3に音素履歴木の例を示す。ルートノード s_0 から各ノードに至る経路が、一つの音素系列を表現する。従って、木のノードを決定することで、音素履歴を一意に決定できる。

音素履歴木の各ノードには、探索部で必要となる、予測音素、後続音素といった、統語解析情報を記録しておく。探索部で音素を検出し、統語情報が必要となる場合に、音素履歴木がまだ構築されていなければ、統語解析の必要があるが、一度構築してしまえば、音素履歴木に対してアクセスするだけでよいので、認識全体における統語解析の負荷は十分小さく抑えることができる。

2 多重サーチの検討

2.1 インクリメンタルサーチの枠組

認識の問題を探索の問題として捉える。適当な拘束条件を満足する解を求めるためには、探索に際して動的に拘束を適用しながら、逐次的に探索領域を更新していく方法が望ましい。しかし、各時点での全ての条件を考慮にいれて条件分岐すると、探索領域は爆発し、総当たり探索は不可能になる。

一般に、各時点での条件分岐に制限を与えることで、逐次的に探索空間を絞り込む。しかし、この方法の最大の問題点は、各時点での条件分岐に与える制限の制御によっては、致命的な誤りを生じることにある。各時点で利用可能な情報は、その時点までの探索結果だけであるため、部分的な探索誤りによっては、それ以降の探索の全てに悪影響を及ぼしかねない。

フレーム同期型の探索手法の最大の問題はこの点である。各フレームまでの尤度に従って仮説を枝刈するために、部分的な照合誤りによる致命的な認識誤りに至る可能性が高い。そうした部分的な誤りによって生じる悪影響を、その前後の探索に対して、極力小さく抑える枠組が必要である。

先に述べたように、探索に適用する拘束を満足し得る仮説展開の組合せによって、探索空間は爆発するため、これを絞り込む必要がある。認識におけるサーチの過程を数段階に分離し、各時点までのサーチの情報とともに、サーチの各段階での探索情報を利用して仮説を絞り込む、多重サーチを考える。サーチの各段階では、言語知識、音声分析結果等の拘束条件を段階的に適用する。サーチのレベルが深まるにつれて、条件分岐数は増加するが、低いレベルでの探索情報を利用することにより、より効率的な探索が可能となり、探索全体での負荷はさほど大きくならないと考えられる。

本研究では、このサーチの過程を前向きサーチと後向きサーチの 2 つの段階に分離する場合について検討する。

前向きサーチと後向きサーチ

前向きサーチは、認識における基本的な探索の過程に対応する。前向きサーチの役割は、後向きサーチでの解析レベルでの探索空間を削減することと、後向きサーチで用いる経路尤度の予備計算である。一方、後向きサーチは、認識における解析の過程に対応する。後向きサーチでは、前向きサーチでの処理結果を受けて、より詳細な解析を行なう。

これまでのフレーム同期型 SSS-LR では、前向きサーチに対する比重が圧倒的に高く、後向きサーチによる解析レベルの処理は、トレースバックのみであった。従って、認識精度の全てが前向きサーチの成功の如何にかかっている。前向きサーチでの一部の探索誤りによって、決定的な認識誤りに至るため、これを回避するために、より安全なビーム幅を設定しなければならず、結果として莫大な計算時間を必要とした。

この前向き・後向きサーチに与える比重をいかに制御するかが、本稿の重要な課題の一つである。

前向きサーチを、後向きサーチに対してどのように位置付けるかによって、その比重は様々に変化する。

- 探索空間を限定するための前向きサーチ
  言語モデルとして文脈自由文法、音響モデルとして文脈依存 HMM を用いているため、全体探索空間は膨大である。この探索空間を総当たり的に網羅することは不可能であるため、前報告では、枝刈の手法により、動的に空間を絞り込んでいた。この探索空間の

削減方法として、前処理的サーチを用いる方法が提案されている [2]。前処理サーチで選択した探索領域のみを、再度、厳密なサーチで詳細に探索し直す。前処理サーチに用いる言語モデル、音響分析法、音響モデル等の拘束を緩めることで、高速化を図る。

- ヒューリスティクス計算のための前向きサーチ
  前向きサーチで計算したスコアを、後向きサーチでのスコア計算に利用する。best-first サーチや A* サーチ等の考え方に従って、各時点までの後向きサーチによって得られる尤度とともに、前向きサーチでの尤度を、それ以降の尤度の近似として用いる。両方向からの尤度に従って、仮説を評価することで、精度の向上を図る。前者と同様、用いる言語モデル、音響分析法、音響モデル等の拘束を緩めて、高速化を図る。

2.2　マージサーチ

マージサーチの概要

厳密に探索を行なう場合には、音素履歴が異なる仮説は音響尤度が異なるため、別々の仮説として扱わなければならない。文中の一部の文節の、さらに一部の助詞だけが異なる複数の文仮説においても、完全に別々の候補として扱わなければならない。しかし、一部を除いて、大半は同じであるといった文仮説における、ほとんどの探索領域は重複している。こうした探索空間の重複を削減できれば、サーチの効率化を図ることができる。

そこで、前向きサーチでの負荷を減じるために、爆発する仮説を極力マージすることを考える。仮説をマージすることにより、展開される仮説数を減少させて、ビームを有効利用できると考えられる。その後、一旦マージされた仮説を後向きサーチにより再構成することで、認識候補を得る。より広いビーム幅を用いる前向きだけのサーチと同等の認識精度が得られると期待されるこのサーチ手法を、以下マージサーチと呼ぶ。

マージは、音素や単語、品詞といった文法カテゴリの様々な単位で可能であり、さらに、文中のカテゴリの位置や、前後のカテゴリによる文脈などの条件下でのマージが考えられる。こうした条件のもとで、どれだけマージが生じるかによって、全体の処理効率に大きく影響するため、できるだけマージが生じやすい条件で前向きサーチを行なって、前向きサーチの負荷を後向きサーチへ分担させ、前向きサーチの安全性をより高めることが望ましい。音素や単語といったより小さな文法カテゴリを単位とすれば、比較的マージが生じやすいが、一方、マージで失われる統語的制約を回復するためには、後向きサーチにおいて、より強い統語制約を用いなければならない。また、より大きな単位でのマージでは、前向きサーチにおけるカテゴリの作成と同時に統語的制約が加味されるものの、マージが生じにくくなるため、前向きサーチの安全性が低くなる。こうしたトレードオフの制御は、インクリメンタルサーチでの本質的な問題である。

本研究では、日本語の文節間の統語的な制約は比較的緩いといった性質から、マージのレベルを文節とする場合が適当であろうと考え、文節レベルでのマージサーチを検討した。

この場合、スポッティングや、progressive サーチにおけるラティスパージングの手法と同様の考え方になる。

マージサーチの概要図を図 1.4に示す。

フレーム同期型 SSS-LR に従って、各フレーム毎に grid 仮説を展開する。grid 仮説の展開の際に、音素間の経路展開によって文節が検出されたならば、その検出フレームを終端とする文節を、その文節の音響尤度とともに、ラティスとしてダンプしていく。フレーム同期型 SSS-LR の終了とともに、後向きサーチによって、文節ラティスを再構築して、認識結果を出力する。

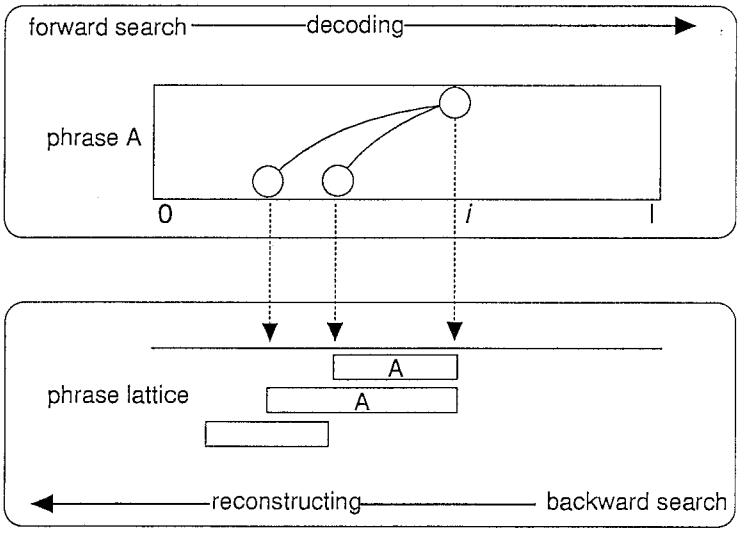

図 1.4: マージサーチの概要

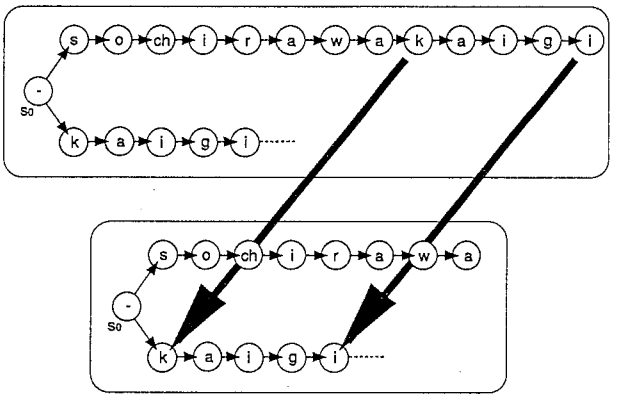

図 1.5: 音素履歴木及び文節内音素履歴木の展開例

前向きサーチ

- 文節内の音素履歴木

前向きサーチでは、照合中の文節が等しい仮説をマージする。従って、仮説の比較の際には、各文節内での音素履歴が等しいか否かを調べることになる。文節内での音素履歴の比較のために、その音素履歴の表現として、文節内音素履歴木を用いる。文節内音素履歴木の構造は、通常の音素履歴木と同じであるが、木の葉に対応するノードが文節の末端に対応し、以降にノードが伸びることはなく、文節を検出した際には、木のルートノード s_0 へとリセットされる。但し、例えば、's o ch i r a d e' という文節に対応する文節内音素履歴木の末端ノードは、's o ch i r a d e w a' という文節の音素履歴木に包含されるため、必ずしも、全ての文節の末端ノードが、木の葉ノードに対応するわけではない。

文節内の音素履歴木と、通常の音素履歴木との対応図を図 1.5に示す。

図 1.5では、文初頭に出現する 'kaigi....' と、'sochirawa' という文節に後続する２番目の文節としての、'kaigi....' がマージされることになる。

- 音素履歴木による曖昧性のパッキングの必要性
  文脈自由文法を用るため、マージの単位を超える統語的な曖昧さが生じる可能性がある。例えば、ｋａｉｇｉとｊｉｍｕｋｙｏｋｕという各々一単語から成る２つの文節と、ｋａｉｇｉｊｉｍｕｋｙｏｋｕという複合単語によって構成される文節では、統語的な曖昧さが生じることになる。この場合、音素履歴は同じであるため音響尤度は等しいので、一つの仮説として扱いたい。そこで、同じ文節であるという条件によってマージするとともに、同じ音素履歴であるか否かによっては、これらを重複しないようにする。従って、文節内の音素履歴木とともに、文の音素履歴木を用いる必要がある。

- マージ単位を超えた制約の適用
  文節間文法などの文節間をまたがる統語制約を用いる場合、文節内音素履歴木ノードが等しい仮説であっても、予測音素 (中心音素) 及び後続音素が同じであるとは限らない。こうしたマージ単位を超えて適用される制約は、前向きサーチで適用することはできない制約であり、後向きサーチにおけるマージ単位の再構築処理において、利用される知識である。従って、より小さな文法カテゴリである単語や品詞をマージ単位とした場合、文節内での統語制約をも欠落するため、さらに後向きサーチにおける負荷は大きくならざるを得ない。

後向きサーチ

後向きサーチでは、前向きサーチで作成した文節ラティスを組合せて文を再構築する。後向きサーチは、従来のラティスパージングの手法に準じるものであり、既に尤度が計算されているラティスの組合せだけの探索空間を調べるだけで済むため、前向きサーチに比べて処理時間は圧倒的に小さくて済む。しかし、前向きサーチのビーム幅を広げると、ラティス数も増加し、探索空間が大きくなるため、処理時間が無視できなくなる可能性がある。そこで、後向きサーチに best-first サーチの技法を用いて、処理の効率化を図る。best-first サーチに基づく、後向きアルゴリズムを以下に示す。

[記号の定義]

I	入力音声の最終フレーム.
P	(前向きサーチで作成された) ラティスの集合.
$<j, i, p>$	入力音声の第 j フレームから、i フレームまでの、文節 p の (前向きサーチで作成された) ラティス ($i = 1 \sim$ I, $j = 1 \sim i$).
(i, l)	入力音声の最終フレームから、第 i フレームまでの、ラティス系列 l についての仮説.
$(<j, i, p> \ l)$	ラティス系列 l に ラティス $<j, i, p>$ を先行結合して作成されるラティス系列.
$c <j, i, p>$	ラティス $<j, i, p>$ の尤度.
$f(i, l)$	ラティス系列仮説 (i, l) のスコア関数.
$g(i, l)$	(i, l) の最終フレームからの累積尤度.
$g(i, l)$	(i, l) の始端フレームまでの推定尤度 $(= h(i))$.
L	(後向きサーチで構築される) ラティス系列仮説の集合.
$Nbest$	Nbest の値.

[ラティス再構築 アルゴリズム]

1. 初期設定
   $f(I,\ ()) = 0$
   $L = \{(I,\ ())\}$
   n =1
   $RecognitionCandidates = NULL$

2. $n > Nbest$ または, $L = \{\}$ ならば,
   $RecognitionCandidates$ を出力し, 終了する.

3. 文節ラティスの展開

   (a) $(i,\ l) = \arg\max_{(I,\ L)\in L} f(I,\ L)$
   $L = L - (i,\ l)$
   $L' = L' + (i,\ l)$

   (b) $i = 1$ ならば,
   - $RecognitionCandidates(n) = l$
   - $n = n + 1$

   (c) $i \neq 1$ ならば,
   $\{< j,\ i',\ p > \mid i' = i\} \in P$ について,
   - $l' = (< j,\ i,\ p >\ l)$
   - $g = g(i,\ l) + c\ < j,\ i',\ p >$
   - $(j,\ l') \notin \{L + L'\}$ ならば,
     $L = L + (j,\ l')$
     $g(j,\ l') = g$
     $f(j,\ l') = g + h(j,\ l')$
   - $(j,\ l') \in L$ かつ $f(j,\ l') < g$ ならば,
     $g(j,\ l') = g$
     $f(j,\ l') = g + h(j,\ l')$

4. 2. に戻る.

best-first サーチに用いる推定スコアは、前向きサーチにより、ラティスの作成に並行して計算することができる。本研究で用いた推定スコアは以下の方法で計算した。

$$h(i,l) = h(i) = h(i-1) + c(i)$$

$$c(i) = \max_{\begin{cases} (i,j,n,s) \in GHYP \\ k = j, j+1 \end{cases}} c < i,j,k,n >$$

ここで、(i,j,n,s) は、前向きサーチでの grid 仮説を表わし、$c < i,j,k,n >$ は、(i,j,n,s) からの HMM の状態遷移による対数確率を表わす $(= log(a^n_{jk}) + log(b^n_{jk}(i)))$。

この推定スコアは、ラティス系列には依存せず、入力音声のフレーム数にのみ依存するため、フレーム数に対する正規化のみの効果をもった推定スコアとなる。また、前向きサーチでビームサーチを用いているため、サーチ全体における最適性は失われているものの、この推定スコアを、上に述べた best-first サーチのアルゴリズムに適用した場合、前向きサーチで作成したラティスの組合せから生じる後向きサーチについては、A* サーチの条件及び単調性の制約条件を満足するため、最適解が保証される。

2.3 ヒューリスティクスサーチ

ヒューリスティクスサーチの概要

多重サーチにおいて、直前の段階までのサーチで計算された尤度を、現段階のサーチに利用することを考える。フレーム同期型 SSS-LR の最大の問題点は、各時刻で仮説を比較する際、仮説の評価関数が各時刻までの仮説の尤度から計算されるために、ある局所的な音響照合の失敗によって致命的な認識誤りを引き起こすということである。一方、音響モデルとして用いる HMnet は比較的信頼性が高く、ビームサーチにおける認識誤りの多くは、ビーム幅を広げれば救うことの出来る仮説、即ち、ビーム幅が小さい場合に枝刈されるものの、本来フルサーチを行なった場合であれば、最終的な尤度は高くなるものが多い。従って、局所的な音響照合の失敗を、できるだけ仮説の評価に影響させない評価関数を用いる事でこれを回避できる。

フレーム同期型のサーチは、各フレームに沿って逐次的に仮説を更新しながら進行するため、サーチの各時点で、それまでの累積尤度によって仮説を評価する。この時、最終的な尤度は高いものの局所的な音響照合の失敗によって正解仮説の尤度が低くなり、小さなビーム幅では、枝刈される可能性がある。しかし、こうした誤認識は、一部の照合の失敗のみに左右されるものであるため、必ずしもサーチ全体において十分広いビーム幅が必要であるわけではない。

こうした観点から、ビーム幅可変のビームサーチ、しきい値制御のビームサーチが検討されている。可変ビーム幅によるビームサーチでは、そのビームの制御が極めて難しい。また、同様に、しきい値制御のビームサーチの場合も、しきい値を一意には決められないため、定量的な手法として確立しているとはいい難い。

音声区間全体において音響照合に失敗する場合は、サーチ方法だけでなく、音響モデルや音声分析のレベルを検討しなければ、認識困難であるが、音声区間の大部分の音響照合は成功するものの、局所的な照合の失敗により認識誤りを生じる場合での、正解を救うことを検討する。仮説の評価方法として、各時刻までの尤度だけでなく、整合度の高い区間での尤度も利用するヒューリスティクスサーチでは、ヒューリスティクス計算のための前処理的サーチ (前向きサーチ) と、ヒューリスティクスを利用した厳密なサーチ (後向きサーチ) で構成される。ヒューリスティクスとして計算される音響尤度を、以下推定スコアと呼ぶ。

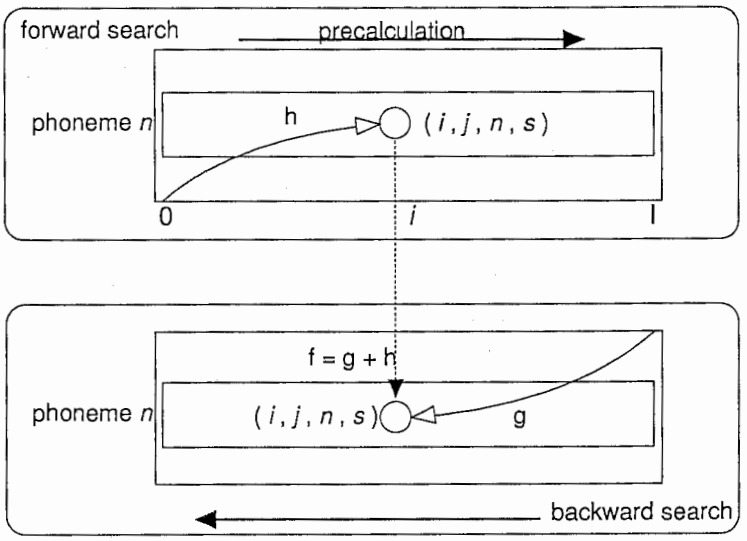

図 1.6: ヒューリスティクスサーチの概要

第 2 章

特定話者文音声認識実験

1 実験条件

実験は、以下の条件のもとで行なった。

音声資料	国際会議に関する問い合わせタスク、136 文、男性話者 MAU。
音響分析	標本化 12kHz、フレーム周期 5 ms、ハミング窓 20 ms。
特徴量	1 〜 16 次 LPC ケプストラム、1 〜 16 次 Δ LPC ケプストラム、log パワー、Δ log パワー。
HMnet	5240 単語の偶数番目の単語から学習、状態数 600。
文脈自由文法	タスク依存型文法 (mset)。マージサーチでは、文節内文法のみを記述した文脈自由文法を用いる。ヒューリスティクスサーチでは、文節内、文節間の両方を考慮した文脈自由文法を用いる。但し、ヒューリスティクス計算のための前向きサーチには, 文脈自由文法から計算した音素対文法を用いる。

2 実験結果

マージサーチによる文音声認識結果を表 2.1 にまとめる。
また表中の文認識率を図 2.1 に示す。但し図 2.1 の縦軸は、文誤認識率である。実線は、1、5、10 位までのマージサーチによる誤認識率、破線は、1、5、10 位までの通常のフレーム同期型 SSS-LR による誤認識率を表わし、マージサーチでの 5、10 の認識結果は等しくなっている。

一文の認識に要した平均 CPU タイムを図 2.2 に示す。各文サンプルの平均リアルタイムは 2025.4 (ms) である。

ヒューリスティクスサーチによる実験結果を表 2.2 にまとめる。表中の文誤認識率を図 2.3 に、一文の認識に要した平均 CPU タイムを図 2.4 に示す。

3 考察

表 2.1及び図 2.1より、マージサーチによる認識率の向上が確認できる。時間の都合上、ビーム幅 200 までの実験しか行なっていないが、さらにビーム幅を広げた場合にその効果が大きく

表 2.1: マージサーチによる各ビーム幅における文認識率

ビーム幅	マージなし			マージあり		
	including			including		
(個)	1	5	10	1	5	10
30	29.4	29.4	29.4	29.4	29.4	29.4
50	44.9	44.9	44.9	44.9	45.6	45.6
100	51.5	52.9	52.9	50.0	54.4	55.2
200	52.9	56.6	56.6	53.7	59.6	60.3

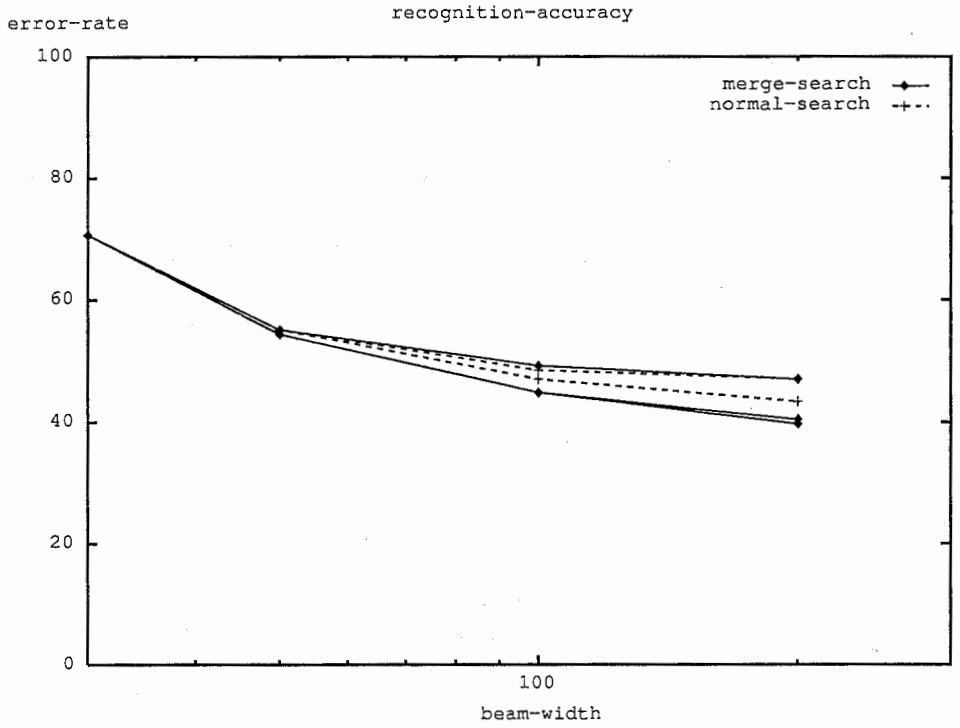

図 2.1: マージサーチにおけるビーム幅 vs. 文誤認識率

表 2.2: ヒューリスティクスサーチによる各ビーム幅における文認識率

ビーム幅	推定スコアなし			推定スコアあり		
	including			including		
(個)	1	5	10	1	5	10
30	36.0	36.0	36.0	61.0	65.4	65.4
50	49.3	49.3	49.3	64.7	69.9	69.9
100	52.9	55.2	55.2	66.9	75.7	75.7
200	58.1	63.2	63.2	68.4	78.7	78.7
400	59.6	66.2	66.2	67.6	78.7	78.7
800	61.8	70.6	70.6	67.6	80.1	80.1

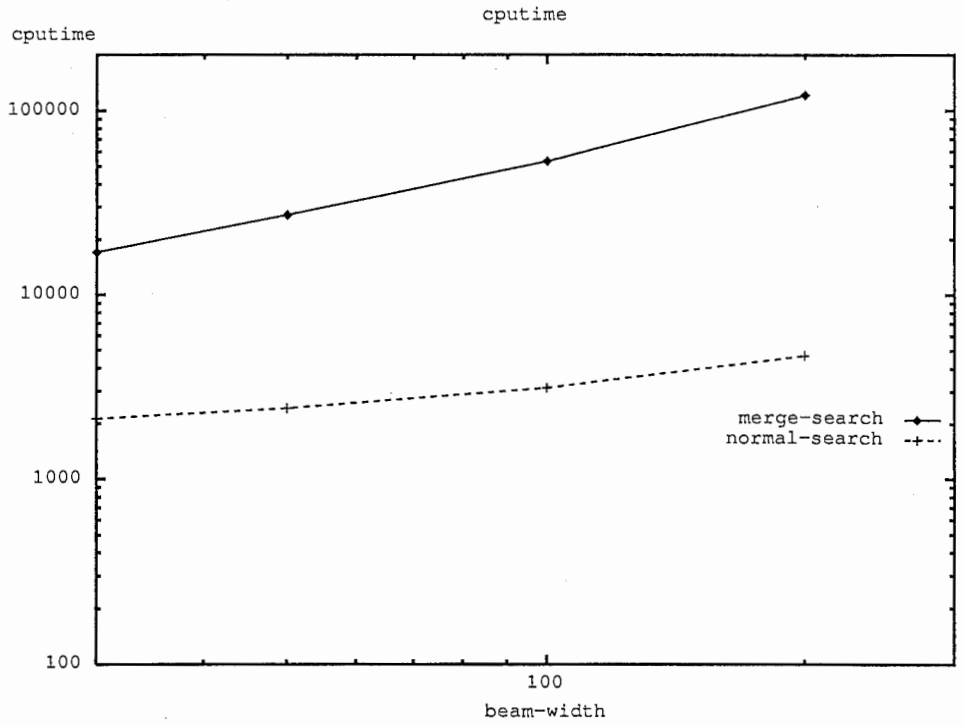

図 2.2: マージサーチにおけるビーム幅 vs. CPU タイム

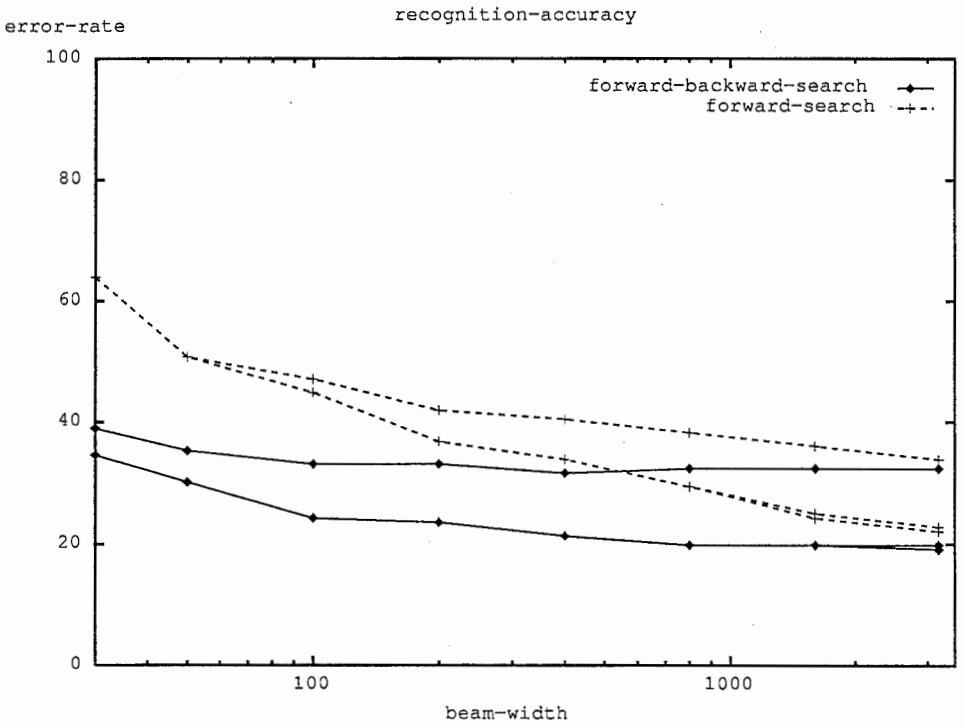

図 2.3: ヒューリスティクスサーチにおけるビーム幅 vs. 文誤認識率

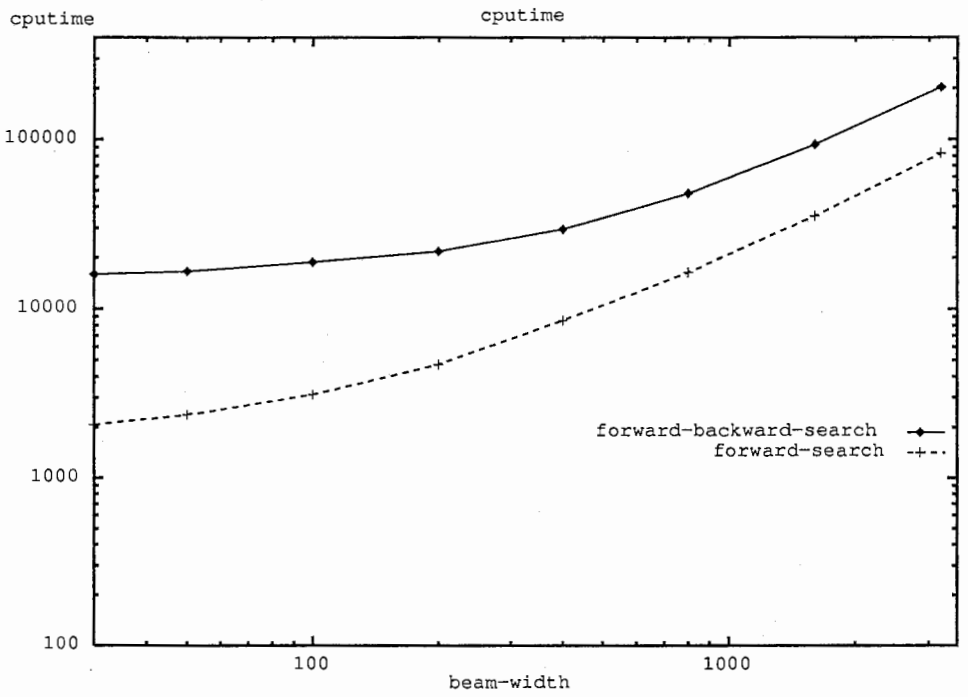

図 2.4: ヒューリスティクスサーチにおけるビーム幅 vs. CPU タイム

なることが予想される。また、best 1 ではさほど顕著な差は見られないが、N-Best を大きくとることによっても、認識率に差が見られ、マージサーチの効果が大きくなることが確認できる。

サーチに要した CPU タイムは、マージしない場合に比べ、マージサーチの場合に圧倒的に大きい。これは、これは、著者のインプリメンテーションが良くなかったためであるが、前向きサーチによって、文節ラティスを形成する際に、各ラティスの開始フレームも同時に保持しながらサーチしたためで、それに伴う処理に莫大な時間を要したことによる。本来はより効率的なマージサーチができるため、プログラムレベルでの改善が早急に必要である。

一方後向きサーチでの消費 CPU タイムは明示していないが、数十 ms から 数百 ms 程度であり、ほぼ無視できることが、以下に示したいくつかの例からも分かる。

また、ビーム幅 100 の best 1 の場合に、マージしない場合の認識率がマージサーチの認識率を上回っている。これは 1 サンプルの認識の差であるが、そのサンプルの認識例を示す。

(認識例 1)

1. 正解文: 会議について、詳しい、事を、教えて下さい。
   – – kaiginitsuite – kuwashii – kotoo – oshietekudasai – –$
2. 実験条件: ビーム幅 100
3. 認識結果 マージあり

```
# Forward Search
Real(Speaking) time: 2840ms ( 568 frames)
CPU(Searching) time: 69990ms (Elapsed. 73sec)
# Backward Search
CPU(Searching) time: 250ms (Elapsed. 0sec)
Recognition Candidates
1 : 228071270( 40.151) q1 pau k a i g i n i t s u i t e pau k u w a s h i i pau k o t o pau o s h i e t e k u d a s a i pau q2 $
```

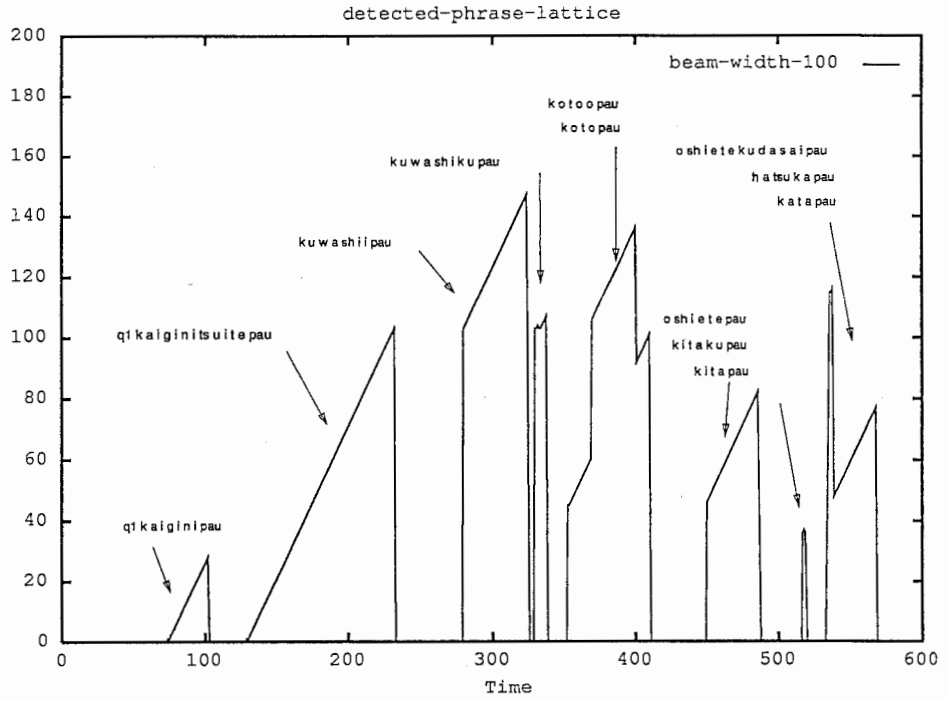

図 2.5: 各フレームにおける検出文節数

```
@ 2 : 228061512( 40.150) q1 pau k a i g i n i t s u i t e pau k u w a sh i i pau k o t o o pau o sh i e t e k u d a s a i pau q2 $
3 : 221937763( 39.072) q1 pau k a i g i n i t s u i t e pau k u w a sh i i pau k o t o pau o sh i e t e pau k a t a pau q2 $
4 : 221928005( 39.070) q1 pau k a i g i n i t s u i t e pau k u w a sh i i pau k o t o o pau o sh i e t e pau k a t a pau q2 $
5 : 221922301( 39.069) q1 pau k a i g i n i t s u i t e pau k u w a sh i i pau k o t o pau o sh i e t e pau h a t s u k a pau q2 $
6 : 221912543( 39.067) q1 pau k a i g i n i t s u i t e pau k u w a sh i i pau k o t o o pau o sh i e t e pau h a t s u k a pau q2 $
```

4. 認識結果 マージなし

```
Recognition Candidates
@ 1 : 228061512(40.150) q1 pau k a i g i n i t s u i t e pau k u w a sh i i pau k o t o o pau o sh i e t e k u d a s a i pau q2 $
Real(Speaking) time: 2840ms (568 frames)
CPU(Searching) time: 4330ms (Elapsed. 8sec)
```

この例は、マージすることにより、よりスコアの高い、即ちより最適解に近い解を出力しているものの、それが不正解であったという例である。

認識結果の @ 印の候補が正解認識で、マージサーチの場合は 2 位で認識し、マージをしない場合には 1 位で認識している。

また、図 2.5 は、このサンプルについての各フレーム毎に検出した文節ラティス数を表している。但し、文節の内容が同じであっても、開始フレームが異なる場合には、別のラティスとして扱っている。例えば、約 100 フレーム付近では、q1 k a i g i n i pau に後続する文節ラティスの数が縦軸になっている。

この他の、ビーム幅 100 における 1 位の認識結果については、マージサーチの場合とマージを行なわないサーチの場合とで同じであった。

マージによって正解が得られた例を以下に示す。

(認識例 2)

1. 正解文: 登録用紙は、既に、お持ちでしょうか。
   — — toorokujooshiwa — sudeni — omochideshjooka — —$

2. 実験条件: ビーム幅 200

3. 認識結果 マージあり

```
# Forward Search
Real(Speaking) time: 2065ms ( 413 frames)
CPU(Searching) time: 63790ms (Elapsed. 64sec)
# Backward Search
CPU(Searching) time: 270ms (Elapsed. 1sec)
Recognition Candidates
1 : 164439312( 39.814) q1 pau t o o r o k u j o o s h i w a pau s u d e n i pau i u pau o m o ch i d e sh j o o k a pau q2 $
2 : 164227049( 39.762) q1 pau t o o r o k u j o o s h i w a pau s u d e n i pau j u u pau o m o ch i d e sh j o o k a pau q2 $
3 : 164216109( 39.760) q1 pau t o o r o k u j o o s h i g a pau s u d e n i pau i u pau o m o ch i d e sh j o o k a pau q2 $
4 : 164003846( 39.708) q1 pau t o o r o k u j o o s h i g a pau s u d e n i pau j u u pau o m o ch i d e sh j o o k a pau q2 $
© 5 : 163956609( 39.697) q1 pau t o o r o k u j o o s h i w a pau s u d e n i pau o m o ch i d e sh j o o k a pau q2 $
6 : 163733406( 39.643) q1 pau t o o r o k u j o o s h i g a pau s u d e n i pau o m o ch i d e sh j o o k a pau q2 $
7 : 163507614( 39.588) q1 pau t o o r o k u j o o s h i j a pau s u d e n i pau i u pau o m o ch i d e sh j o o k a pau q2 $
8 : 163481145( 39.582) q1 pau t o o r o k u j o o s h i n i w a pau s u d e n i pau i u pau o m o ch i d e sh j o o k a pau q2 $
9 : 163444229( 39.573) q1 pau t o o r o k u j o o s h i w a pau s u d e n i pau n i pau i u pau o m o ch i d e sh j o o k a pau q2 $
10 : 163339831( 39.548) q1 pau t o o r o k u j o o s h i w a pau s o r e n i pau i u pau o m o ch i d e sh j o o k a pau q2 $
```

4. 認識結果 マージなし

```
Recognition Candidates
1 : 164439312(39.814) : q1 pau t o o r o k u j o o s h i w a pau s u d e n i pau i u pau o m o ch i d e sh j o o k a pau q2 $
Real(Speaking) time: 2065ms (413 frames)
CPU(Searching) time: 4850ms (Elapsed. 6sec)
```

この例では、マージなしの場合に、3 番目の文節と 4 番目の文節の間に、文節が挿入されたために誤認識となったものを、マージサーチでは、後向きサーチによるラティスの再構築によって、5 位で正解を救った例である。

(認識例 3)

1. 正解文: 登録用紙で、手続きを、して下さい。
   – – toorokujooshide – tetsuzukio – shitekudasai – –$

2. 実験条件: ビーム幅 200

3. 認識結果 マージあり

```
# Forward Search
Real(Speaking) time: 2080ms ( 416 frames)
CPU(Searching) time: 58570ms (Elapsed. 59sec)
# Backward Search
CPU(Searching) time: 130ms (Elapsed. 0sec)
Recognition Candidates
© 1 : 156445682( 37.605) q1 pau t o o r o k u j o o s h i d e pau t e ts u z u k i o pau sh i t e k u d a s a i pau q2 $
2 : 156357685( 37.584) q1 pau t o o r o k u j o o s h i d e pau t e ts u z u k i w a pau sh i t e k u d a s a i pau q2 $
3 : 156196732( 37.545) q1 pau t o o r o k u j o o s h i d e pau t e ts u z u k i n o pau sh i t e k u d a s a i pau q2 $
4 : 156026551( 37.505) q1 pau t o o r o k u j o o s h i d e pau t e ts u z u k i g a pau sh i t e k u d a s a i pau q2 $
5 : 153945159( 37.004) q1 pau t o o r o k u j o o s h i d e pau t e ts u z u k i o pau sh i t e pau k a t a pau s a N pau q2 $
6 : 153857162( 36.983) q1 pau t o o r o k u j o o s h i d e pau t e ts u z u k i w a pau sh i t e pau k a t a pau s a N pau q2 $
7 : 153696209( 36.944) q1 pau t o o r o k u j o o s h i d e pau t e ts u z u k i n o pau sh i t e pau k a t a pau s a N pau q2 $
8 : 153526028( 36.903) q1 pau t o o r o k u j o o s h i d e pau t e ts u z u k i g a pau sh i t e pau k a t a pau s a N pau q2 $
9 : 152888910( 36.750) q1 pau t o o r o k u j o o s h i d e pau t e ts u z u k i e pau sh i t e k u d a s a i pau q2 $
10 : 152568556( 36.673) q1 pau t o o r o k u j o o s h i d e pau t e ts u z u k i o pau sh i t e pau k u g a ts u pau s a N pau q2 $
```

4. 認識結果 マージなし

```
Recognition Candidates
1 : 154447429(37.125) : q1 pau t o o r o k u j o o s h i d e pau t e ts u z u k i o pau k e q k a g a pau s a ng pau q2 $
Real(Speaking) time: 2080ms (416 frames)
CPU(Searching) time: 4860ms (Elapsed. 9sec)
```

この例では、マージをしない場合には、サーチの途中で 3 番目の文節が枝刈されたが、マージサーチでは、3 番目の文節ラティスの形成に成功したことにより、最終的に正解認識が得られたという場合の例である。

全体を通して、3 番目の例のようなマージによるビームの有効利用の効果よりも、2 番目の例のように、文節ラティスを検出した後の、後向きサーチによって正解を救うという、効果の方が大きい。逆にいえば、前向きサーチにおいて、正解文節ラティスの検出に失敗した場合は、後向きサーチで救うことができなくなり、致命的である。従って、前向きサーチにおける

安全性が重要となり、マージによるビームの有効利用の効果だけでなく、同時に、何らかのより安全な前向きサーチを検討しなければならない。

また、図 2.3 より、ヒューリスティクスサーチでは、ビーム幅を 100 から 200 程度の場合に、ビーム幅 3200 までの通常の前向きサーチと同等の認識率が得られる。一方、ビーム幅 3200 の前向きサーチに必要な CPU タイムは、ビーム幅を 800 から 1600 程度のヒューリスティクスサーチと同等である。

前向きサーチとしての、音素対文法 (FSA) 制御の One Pass Viterbi では、フルサーチによって推定スコアを設定するため、処理時間が大きい。ビーム幅を同等として比較する場合、前向きサーチの処理時間による差が顕著に表われる。

以上の実験結果を総じていえることは、

- マージによるビームの有効利用は、顕著とは言えないながらも効果がある。マージによって前向きサーチの負荷を、後向きサーチに分担させることにより、認識精度の向上の可能性がある。
- ヒューリスティクスサーチの効果はある。特に、小さなビーム幅での認識率を向上させることが可能となる。
- マージサーチにおいて、検出文節を再構築することにで、N 位までに正解を得られる可能性が高くなり、認識精度を向上できる。
- マージサーチの効果として、ビームの有効利用の効果より、文節の再構築の効果の方が、認識率の向上に対する貢献度が大きい。
- 前向きサーチでのビーム幅を広げることに比べ、ヒューリスティクスサーチによって認識精度を向上させる方が、認識全体のパフォーマンスは高く、特に、小さなビーム幅でのヒューリスティクスサーチで顕著に見られる。
- 最適な解を得ることが必ずしも認識率の向上につながるとはいえず、特に音響モデルの信頼性には、大きく依存せざるを得ない。従って、音響モデルの信頼性を考慮に入れたより頑健な手法としての見直しが必要である。

第 3 章

むすび

本研究では、フレーム同期型 SSS-LR による文音声認識について検討した。文音声認識における処理効率の向上を目的として、多重サーチを検討した。マージサーチ、ヒューリスティクスサーチについて、認識実験を行ない、認識率と計算量について評価を行なった。実験結果から、こうした多重サーチによる処理効率の向上が期待できることが確認された。

今後の検討課題として、

1. アルゴリズム、インプリメンテーションレベルでの高速化の必要がある。特にマージサーチでは、著者のプログラミングの問題が大きく、高速化が必要である。

2. マージサーチにおける、マージのカテゴリについての検討の余地がある。本実験では、マージのカテゴリとして文節のみを検討したが、本来は種々のカテゴリについて幅広く実験を行なうべきである。

3. ヒューリスティクスサーチにおいて、前向きサーチにおいても枝刈を用いて処理量を低減させるなどの、推定スコア設定のための前向きサーチの計算時間をさらに短縮させる必要がある。

4. 前向き・後向きサーチだけでなく、さらにサーチの段階数を増やすことで、全体のパフォーマンスを向上させる検討が考えられる。

5. 特にヒューリスティクスサーチでは、音声入力終了後に比較的計算量の大きな処理 (後向きのビームサーチ) を行なうため、リアルタイム音声認識システムの構築が難しい。従って、音声区間の全体に対するサーチとは別に、複数の部分区間のサーチによって全体を認識する等、サーチの枠組を検討する必要がある。

6. 定型的な発話音声データに対する認識パフォーマンスによってだけでなく、特に自由発話データに対しての、認識手法としてのフレーム同期型 SSS-LR を、総合的に評価し、問題点を明確にする必要がある。本実験では、音響モデルとして用いる HMnet は、入力音声データに対して比較的信頼性が高く、また、文脈自由文法は発話内容に完全に適合することを前提とし、その条件下での純粋なサーチの問題としてのみ検討を行なった。しかし、自由発話データでは、この前提が保証されないため、十分な実験とともに、検討する必要がある。

謝辞

本実習の機会を与えていただいた、山形大学 好田 正紀 教授、ATR 音声翻訳通信研究所 山崎 泰弘 社長に感謝致します。

また、本実習に当たり、懇切丁寧な御指導と、厚い激励をいただきました、清水 徹 研究員、松永 昭一 主任研究員、匂坂 芳典 第一研究室室長に感謝致します。

最後に、Harald Singer 研究員、東北大学 中井 満氏 他 ATR 音声翻訳通信研究所の皆様には貴重な意見とともに、惜しみない技術的指導をいただきました。深く感謝致します。

参考文献

[1] A. Nagai, K. Kita, and S. Sagayama. HMM-LR 法における音素文脈依存型 LR パーザの検討. 音講論, pages 125–126, September 1990.

[2] R.Schwartz and Y.L.Chow. THE N-BEST ALGORITHM:an efficient and exact procedure for finding the N most likely sentence hypotheses. *Proc. ICASSP*, (S2.12):81–84, 1990.

[3] 嵯峨山 高見. 逐次状態分割法 (SSS) による隠れマルコフネットワークの自動生成. 音講論, (2-5-13), 1991.

[4] 岡田 美智男. 文脈自由な句構造文法からのミニマル・サブネットワークの生成について. 信学技報, (SP91-27), June 1991.

付録 A

フレーム同期型 SSS-LR の仕様及びソースコードについて

1 仕様及びソースコード

フレーム同期型 SSS-LR に関して、1994 年春季、1994 年夏季、1995 年春季の 3 度に渡り、ソースコードを変更し、またその仕様も少しずつ変更してきました。

そこで、本テクニカルレポートの提出にあたって、現時点でのソースコード及び仕様についてまとめる。

1.1 仕様

Version を特に区別する必要がない限り、フレーム同期型 SSS-LR を FS-SSS-LR と、音素同期型 SSS-LR を PS-SSS-LR と称す。

PS-SSS-LR[1]を改良し FS-SSS-LR を作成する。その仕様はほぼ音素同期型 SSS-LR と同様で、 VERSION3 までで以下の通り。

- 確率テーブルの読み込み
  makesample0.awk、expand_sample、WaveParam、cal_frame_prob
  で作成された確率テーブルを stdin から読み込む。
  但し、cal_frame_prob で用いる HMnet と FS-SSS-LR.exe で用いる HMnet は同じものである。

- その他、CFG file、LR テーブル file、HMnet file、duration file、gramod file などは、PS-SSS-LR.exe と同じ仕様。

以下の点は、VERSION3 までに変更した機能である。

- phrase by phrase と同時に frame by frame での入力が可能。
- 認識結果を日本語で表示可能。
- 認識結果のトレースバック出力の強化。
- rpc(remote procedure call) 出力が可能。

但し、認識出力結果を日本語で表示するため、生成規則 - 仮名漢字対応辞書が必要になります。

[1]version は不明、singer さんからいただいたもの

1.2 ソースコード

ソースコードのコンパイルや動作確認は原則として HP-UX のみで行なっています。従って、DEC ultrix、alpha や SunOS 等でのコンパイルが正常に動作するかどうかは不明。

PS-SSS-LR のほぼ super set になっていますが、PS-SSS-LR のソースコードの内、データファイルの読み込みや構造体の作成などの機能しか共有していない。

One-Pass Viterbi に纏わる機能と、文脈依存 LR パーサなどは、新たに作成。

VERSION2 から VERSION3 に更新する際、PS-SSS-LR にのみ必要で、FS-SSS-LR には必要のないファイルをいくつか削除。また全てのソースファイルは C++ コンパイラ (gcc v2.6) で通るように変更。但しオプションを解析する部分を除き、ほとんどの部分は ANSI C 準拠に変更したのみです。従って特にプロトタイプ宣言を統一しました。また、rpc が C++ をサポートしていないのかどうか分かりませんが、rpc コードがコンパイルできませんでしたので、ANSI C への変更とともに C linkage を用いた。但しこれは一時的な処理であり、本来 rpc の C++ version があればすげかえる必要がある。

ソースファイルとディレクトリ

ソースファイルは、

x̃monzen/Work3/src/3.0/3.00

にあります。コンパイルには make を使う。詳細は直接 Makefile を見て下さい。

1.3 その他

動作確認

VERSION3 に関する動作確認のため、
x̃monzen/Work3/src/3.0/MikoshiSample2
に、テストサンプル[2]があります。この directory にあるＣＦＧファイル mset_phrase.sss.gra の一部と、テスト用の実行スクリプト、
x̃monzen/Work3/src/3.0/3.00/.csh
を以下に示す。

```
    #!/bin/csh -f
#--------------------
set DIR = ../MikoshiSample2
set LOG = FS-SSS-LR.$OS.log
if (-e $LOG) /bin/rm -f $LOG
#--------------------
cat $DIR/para.binary| \
./FS-SSS-LR.$OS.exe \
-b 64 \
-B 100 \
-g $DIR/mset_phrase.sss \
-l $DIR/gra2mod.min \
-L $DIR/gra2dur.min \
-m $DIR/HMnet.F.201.retrain.150 \
-n 10 \
-P 5 \
-D $DIR/HMnet_filled.100 \
-fpt 2 \
-fmt 2 \
-fdt 0 \
-fphraseunit \
|& tee -a $LOG
```

[2]singer さんの" みこし" デモ用のサンプルデータ

```
--------Context Free Grammar--------
(< start >< -- > (< _start >))
(< bunsetu >< -- > (< np >))
(< bunsetu >< -- > (< wh - np >))
(< bunsetu >< -- > (< adv - ph >))
(< bunsetu >< -- > (< np - no >))
(< bunsetu >< -- > (< adv - p - no >))
(< bunsetu >< -- > (< pre - v - ni >))
(< bunsetu >< -- > (< wh - np - e >))
(< bunsetu >< -- > (< n - keisiki - vaux - cop >))
(< bunsetu >< -- > (< n - keisiki - vaux - coord >))
(< bunsetu >< -- > (< n - keisiki - vaux - katei >))
(< bunsetu >< -- > (< n - keisiki - vaux - s + f >))
(< bunsetu >< -- > (< np - keisiki >))
(< bunsetu >< -- > (< np - keisiki - e >))
(< bunsetu >< -- > (< n - vaux - cop >))
(< bunsetu >< -- > (< n - vaux - coord >))
(< bunsetu >< -- > (< n - vaux - katei >))
(< bunsetu >< -- > (< n - vaux - s + f >))
(< bunsetu >< -- > (< v - no - p >))
(< bunsetu >< -- > (< n - ren - p - k >))
(< bunsetu >< -- > (< nomina - sa >))
(< bunsetu >< -- > (< vaux - sa - nom >))
(< bunsetu >< -- > (< vaux - sa - te >))
(< bunsetu >< -- > (< vaux - sa - h >))
(< bunsetu >< -- > (< vaux - naru - s >))
(< adv - s1 >< -- > (doosureba))
(< vk - s1 >< -- > (joroshiidesuka))
(< mondai >< -- > (moNdaiarimaseN))
(< ka - douka >< -- > (< vaux - sfp - ka >< adv - p - f >))
(< adv - p - f >< -- > (< adv - k - d >< p - f - 2 >))
(< bunsetu >< -- > (< conj >))
(< bunsetu >< -- > (< vaux - cop >))
(< bunsetu >< -- > (< vaux - coord >))
(< bunsetu >< -- > (< vaux - h >))
(< bunsetu >< -- > (< vaux - s >))
(< bunsetu >< -- > (< vaux - sfp >))
(< bunsetu >< -- > (< vaux - te >))
(< bunsetu >< -- > (< vaux - nom >))
(< bunsetu >< -- > (< n - rentai >))
(< bunsetu >< -- > (< adre - vaux - cop >))
(< bunsetu >< -- > (< adre - si >))
(< bunsetu >< -- > (< adre - tyo >))
(< bunsetu >< -- > (< adre - num - ch >))
(< bunsetu >< -- > (< adre - num >))
(< bunsetu >< -- > (< adre - num - np - o >))
(< bunsetu >< -- > (< tel - vaux - cop >))
(< bunsetu >< -- > (< money - man >))
(< bunsetu >< -- > (< money - hyaku >))
(< bunsetu >< -- > (< n - vaux - money - sen >))
(< bunsetu >< -- > (< n - vaux - cop - money - man >))
(< bunsetu >< -- > (< n - vaux - cop - money - hyaku >))
(< bunsetu >< -- > (< n - vaux - coord - money - sen >))
(< bunsetu >< -- > (< n - vaux - katei - money - man >))
(< bunsetu >< -- > (< n - vaux - katei - money - hyaku >))
(< bunsetu >< -- > (< n - vaux - s - money - sen >))
(< bunsetu >< -- > (< np - money - man >))
(< bunsetu >< -- > (< np - money - hyaku >))
(< bunsetu >< -- > (< np - no - money - sen >))
(< bunsetu >< -- > (< np - e - money - man >))
(< bunsetu >< -- > (< np - e - money - hyaku >))
(< bunsetu >< -- > (< n - vaux - h - money - sen >))
(< bunsetu >< -- > (< n - quote - money - man >))
(< bunsetu >< -- > (< n - quote - money - hyaku >))
(< bunsetu >< -- > (< n - vaux - s + f - money - sen >))
(< bunsetu >< -- > (< n - vaux - sfp - money - man >))
(< bunsetu >< -- > (< n - vaux - sfp - money - hyaku >))
(< bunsetu >< -- > (< np - o >))
(< bunsetu >< -- > (< np - money - sen - o >))
(< bunsetu >< -- > (< wh - np - o >))
(< bunsetu >< -- > (< v - no - p - o >))
(< bunsetu >< -- > (< vaux - sa - s + f >))
(< bunsetu >< -- > (< vaux - adj >))
(< bunsetu >< -- > (< vaux - adj - sfpq >))
(< adre - vaux - cop >< -- > (< adre - num - gou >< v - cop >))
(< adre - num - np >< -- > (< adre - num - gou >< p >))
(< tel - vaux - cop >< -- > (< tel - 4 >< v - cop >))

(< tel - no >< -- > (< tel - 3 >< p - kaku3 >))

(< n - vaux - cop >< -- > (< n - nomina - cop >< v - cop >))
(< n - vaux - cop - money - man >< -- > (< n - money - man >< v - cop >))
(< n - vaux - cop - money - hyaku >< -- > (< n - money - hyaku >< v - cop >))
(< n - vaux - cop >< -- > (< n >< p - kaku5 >< v - cop >))
(< n - quote >< -- > (< n - vaux - cop >< p - i >))
(< n - vaux - s >< -- > (< n - vaux - cop >< sp - rentai >))

(< n - vaux - sfp >< -- > (< n - vaux - cop >< sfp >))
(< n - nomina >< -- > (< n >< cop - rentai >))
(< v - cop >< -- > (< v - cop - mizen >< intn >))
(< vaux - coord >< -- > (< v - 5dan - i >))
(< vaux - coord >< -- > (< kyo - renyo1 >))

(< vaux - s >< -- > (< v - renyo2 - t >< sp - renyo - t - all >))

(< vaux - naru - s >< -- > (< pre - v - sahen >< p - kaku - ni >< dont - naru - i >< masu - rentai >< sp - rentai >))
(< vaux - s >< -- > (< v - 5dan - i >< masu - renyo >< sp - renyo - t - all >))
(< vaux - sa - s >< -- > (< v - sa - katei >< sp - katei >))

(< _start >< -- > (q1 < bunsetu > q2))
(< bunsetu >< -- > (< np - special >))
(< bunsetu >< -- > (< wh - n >))
(< bunsetu >< -- > (< adv - p >))
(< bunsetu >< -- > (< wh - np - no >))
(< bunsetu >< -- > (< vn - no >))
(< bunsetu >< -- > (< np - e >))
(< bunsetu >< -- > (< n - keisiki - vaux >))
(< bunsetu >< -- > (< n - keisiki - quote >))
(< bunsetu >< -- > (< n - keisiki - vaux - h >))
(< bunsetu >< -- > (< n - keisiki - vaux - s >))
(< bunsetu >< -- > (< n - keisiki - vaux - sfp >))
(< bunsetu >< -- > (< np - keisiki - no >))
(< bunsetu >< -- > (< n - vaux >))
(< bunsetu >< -- > (< n - quote >))
(< bunsetu >< -- > (< n - vaux - h >))
(< bunsetu >< -- > (< n - vaux - s >))
(< bunsetu >< -- > (< n - vaux - sfp >))
(< bunsetu >< -- > (< n - ren - p >))
(< bunsetu >< -- > (< vaux - sa >))
(< bunsetu >< -- > (< vaux - sa - s >))
(< bunsetu >< -- > (< vaux - sa - coord >))
(< bunsetu >< -- > (< vaux - sa - mod >))
(< bunsetu >< -- > (< vaux - naru >))
(< bunsetu >< -- > (< adv - s1 >))
(< bunsetu >< -- > (< vk - s1 >))
(< bunsetu >< -- > (< mondai >))
(< bunsetu >< -- > (< ka - douka >))
(< vaux - sfp - ka >< -- > (< vaux >< sfp - q >))
(< bunsetu >< -- > (< interj >))
(< bunsetu >< -- > (< vaux >))
(< bunsetu >< -- > (< vaux - sa - cop >))
(< bunsetu >< -- > (< vaux - kdo - coord >))
(< bunsetu >< -- > (< vaux - katei >))
(< bunsetu >< -- > (< vaux - s + f >))
(< bunsetu >< -- > (< vaux - sa - sfp >))
(< bunsetu >< -- > (< vaux - mod >))
(< bunsetu >< -- > (< quote >))
(< bunsetu >< -- > (< vaux - ni >))
(< bunsetu >< -- > (< adre - ken >))
(< bunsetu >< -- > (< adre - ku >))
(< bunsetu >< -- > (< adre - no >))
(< bunsetu >< -- > (< adre - num - ban >))
(< bunsetu >< -- > (< adre - num - np >))
(< bunsetu >< -- > (< adre - num - np - e >))
(< bunsetu >< -- > (< tel - no >))
(< bunsetu >< -- > (< money - sen >))
(< bunsetu >< -- > (< n - vaux - money - man >))
(< bunsetu >< -- > (< n - vaux - money - hyaku >))
(< bunsetu >< -- > (< n - vaux - cop - money - sen >))
(< bunsetu >< -- > (< n - vaux - coord - money - man >))
(< bunsetu >< -- > (< n - vaux - coord - money - hyaku >))
(< bunsetu >< -- > (< n - vaux - katei - money - sen >))
(< bunsetu >< -- > (< n - vaux - s - money - man >))
(< bunsetu >< -- > (< n - vaux - s - money - hyaku >))
(< bunsetu >< -- > (< np - money - sen >))
(< bunsetu >< -- > (< np - no - money - man >))
(< bunsetu >< -- > (< np - no - money - hyaku >))
(< bunsetu >< -- > (< np - e - money - sen >))
(< bunsetu >< -- > (< n - vaux - h - money - man >))
(< bunsetu >< -- > (< n - vaux - h - money - hyaku >))
(< bunsetu >< -- > (< n - quote - money - sen >))
(< bunsetu >< -- > (< n - vaux - s + f - money - man >))
(< bunsetu >< -- > (< n - vaux - s + f - money - hyaku >))
(< bunsetu >< -- > (< n - vaux - sfp - money - sen >))
(< bunsetu >< -- > (< n - ren - p - o >))
(< bunsetu >< -- > (< np - money - man - o >))
(< bunsetu >< -- > (< np - money - hyaku - o >))
(< bunsetu >< -- > (< np - keisiki - o >))
(< bunsetu >< -- > (< vaux - naru - s + f >))
(< bunsetu >< -- > (< nomina >))
(< bunsetu >< -- > (< vaux - adj - sfp >))
(< adre - vaux - cop >< -- > (< adre - num >< v - cop >))
(< adre - no >< -- > (< adre - num >< p - kaku3 >))

(< adre-num-np >< -- > (< adre-num-gou >< wh-p >))
(< tel - vaux - cop >< -- > (< tel - 4 >< cop - renyo1 >< polt - v - rentai >< sfp >))
(< n - vaux >< -- > (< n >< cop - renyo1 >< polt - v - rentai >))
(< n - vaux - cop >< -- > (< n >< v - cop >))
(< n-vaux-cop-money-sen >< -- > (< n-money-sen >< v - cop >))
(< n-keisiki-vaux-cop >< -- > (< n-keisiki >< v-cop >))
(< n - vaux - cop >< -- > (< wh - n >< v - cop >))
(< n - vaux - coord >< -- > (< n >< cop - renyo1 >))
(< n-vaux-s+f >< -- > (< n-vaux-cop >< sp-rentai+f >))
(< n - nomina - cop >< -- > (< n - nomina >< p - j >))
(< vaux - cop >< -- > (< adv >< v - cop >))
(< v - cop >< -- > (< v - cop - rentai >))
(< vaux - kdo - coord >< -- > (< kdo - renyo3 >))
(< vaux-s >< -- > (< nomina-cop >< v-cop-renyo2 >< sp - renyo - t - 2 >))
(< vaux - sa - s >< -- > (< v - sa - renyo2 - t >< sp - renyo - t - all >))
(< vaux - s >< -- > (< dont - 5dan - i - ken >< masu - renyo >< sp - renyo - t - all >))
(< vaux - s >< -- > (< polt - v - renyo2 - t >< sp - renyo - t - all >))
(< vaux - s >< -- > (< v - 5dan - e >< sp - katei >))
```

実行結果は、以下のように出力される。

```
    This is FS-SSS-LR.HP-UX.exe, Version 3.0
<1995/04/13 by xmonzen>
HMnet -> ../MikoshiSample2/HMnet.F.201.retrain.150
HMnet ( model ) State num = 201

Duration -> ../MikoshiSample2/HMnet_filled.100
Duration num = 100

Nsd = 1.000000
Dur_Break_point = 0.000000
Dur_Rate = 0.000000

allophone_num:552 ...
allophone_num:100 ...
-> Loading grammar rules from '../MikoshiSample2/mset_phrase.sss.gra'...Ok [979 rules]
-> Loading LR parsing table from '../MikoshiSample2/mset_phrase.sss.tab'...Ok [2247 states]
-> Padding Phoneme dictionary (27)
-> Loading kana-kanji table from '../MikoshiSample2/mset_phrase.sss.dic'... Ok.
-> NORMALIZE BY NUM-WORDS.
-> W1(GrammarWeight) = 0.000000
-> W2(SyllableWeight) = 0.000000
-> W3(RuleWeight) = 0.000000
MESSAGE (FS-SSSLR): Expanding HMnet....ok(state = 2134)
Initialization completed: 9590secCPU 10secElaps
MESSAGE (FS-SSSLR): will trace back
MESSAGE (FS-SSSLR): byte = 106932
MESSAGE (FS-SSSLR): frame = 133 state = 201
MESSAGE (FS-SSSLR): One-Pass Viterbi SSS-LR Runs(width 100:64)
..... ..... .....
..... ..... .....
..... ..... .....
..... ..... .....
..... ...

[RecognitionCandidates]
1 : 47319940(35.577) : - m o sh i m o sh i - $
2 : 44308130(33.313) : - o sh i h a r a i sh i t e - $
[Real(Speaking)time] : 665ms(133frames)
[CPU(Searching)time] : 730ms(Elapsed.1sec)

#phoneme-trace
[1]
```

```
    0   -     7    -
    8   -    17   m
   18   -    27    o
   28   -    42   sh
   43   -    47    i
   48   -    59    m   [2]
   60   -    65    o
   66   -    84   sh
   85   -   126    i
  127   -   132    -
  133   -   133    $
    0   -     7    -
    8   -    27    o
   28   -    42   sh
   43   -    47    i
   48   -    50    h
   51   -    54    a
   55   -    58    r
   59   -    62    a
   63   -    65    i
   66   -    84   sh
   85   -   124    i
  125   -   127    t
  128   -   131    e
  132   -   132    -
  133   -   133    $
#morpheme-trace

[1]
    0   -     7      ポーズ
    8   -   126    もしもし
  127   -   133      ポーズ
[2]
    0   -     7      ポーズ
    8   -    65    お支払い
   66   -   124          し
  125   -   131          て
  132   -   133      ポーズ
MESSAGE (FS-SSSLR): byte = 234768

MESSAGE (FS-SSSLR): frame = 292 state = 201

MESSAGE (FS-SSSLR): One-Pass Viterbi SSS-LR Runs(width 100:64)

..... ..... .....

..... ..... .....

..... ..... .....

..... ..... .....

..... ..... .....

..... ..... .....

..... ..... .....

..... ..... .....

..... ..... .....

..... ..... ..
```

[*RecognitionCandidates*]

1 : 97293283(33.318) : - k o ng d o d e sh j o o t o - $

2 : 90551547(31.009) : - k o ng d o d e sh j o o k a - $

3 : 90547983(31.008) : - k o ng d o d e sh j o o g a - $

4 : 88951132(30.461) : - k o ng d o d e g o z a i m a s u - $

5 : 87431618(29.941) : - k o ng d o d e sh j o o n o d e - $

6 : 85774718(29.373) : - k o ng d o e n o - $

7 : 85600575(29.314) : - k o ng d o d e n o - $

8 : 84188135(28.830) : - k o ng d o d e s u n o d e - $

9 : 70237980(24.053) : - k o ng d o d e sh j o o n e - $

10 : 59360558(20.328) : - k o ng d o d e sh j o o - $

[*Real(Speaking)time*] : 1460*ms*(292*frames*)

[*CPU(Searching)time*] : 1650*ms*(*Elapsed.2sec*)

```
#phoneme-trace

[1]
    0   -    6    -
    7   -   17    k
   18   -   27    o
   28   -   32    ng
   33   -   36    d
   37   -   57    o
   58   -   61    d
   62   -  111    e
  112   -  115    sh
  116   -  129    j
  130   -  134    o
  135   -  201    o
  202   -  220    t
  221   -  285    o
  286   -  291    -
  292   -  292    $
[2]
    0   -    6    -
    7   -   17    k
   18   -   27    o
   28   -   32    ng
   33   -   36    d
   37   -   57    o
   58   -   61    d
   62   -  111    e
  112   -  115    sh
  116   -  129    j
  130   -  134    o
  135   -  200    o
  201   -  221    k
  222   -  290    a
  291   -  291    -
  292   -  292    $
#morpheme-trace

[1]
    0   -    6    ポーズ
    7   -   57      今度
   58   -  134    でしょ
  135   -  201       う
  202   -  285       と
  286   -  292    ポーズ
[2]
    0   -    6    ポーズ
    7   -   57      今度
   58   -  134    でしょ
  135   -  200       う
  201   -  290       か
  291   -  292    ポーズ
```