ＴＲ－ＳＬＴ－００１４

Bag トランスレーション実験報告

Bag Translation Revisited

谷田 泰郎†　　田中 英輝
Yasuo Tanida and Hideki Tanaka

2002 年 6 月 14 日

概要

本報告では ATR が保有するデータベースの内、NHK ニュース（日、英）、フレーズブック（日、英）、日経新聞（英）を対象に行った N グラムによる単語の並び替え実験について報告する。本報告では実験アルゴリズムの詳細を説明し、基本的な実験結果を報告する。次に、文長や探索の枝刈りが正解率や処理時間に及ぼす影響を議論する。基本的な実験では文長 10 の 1000 文を対象に、並べ替えを全解探索した。この結果、3 グラムを使った場合、NHK 日本語ニュースで 33%、同英語ニュースで 25%、フレーズブック日本語データでは 42%、英語データでは 38%の文を完全に復元できた。またフレーズブックの平均的な長さの文であれば 70%近い正解率を得ることができた。さらに、高速化のための枝刈り実験によると、かなりの枝刈りを行っても完全な探索と同等の正解率になることが分かった。

（株）国際電気通信基礎技術研究所
音声言語コミュニケーション研究所
〒619－0288 京都府相楽郡精華町光台二丁目 2 番地 2　TEL：0774－95－1301
†ＴＩＳ株式会社
Advanced Telecommunication Research Institute International
Spoken Language Translation Research Laboratories
2-2-2 Hikaridai Seika-cho Soraku-gun Kyoto 619-0288,Japan
Telephone:+81-774-95-1301
Fax　　:+81-774-95-1308

目次

図一覧

表一覧

1 はじめに

1990 年、Brown 等が統計翻訳システムを提案した論文 “A Statistical Approach to Machine Translation”に”bag translation”と呼ばれる実験が報告されている(Brown 90)。この実験は次のような手順に従ったものである。

1) 英文を単語に区切って袋（bag）に入れる
2) 袋の中の単語を取り出して、可能な文すべてを作成する
3) 各文の出現確率を N—gram で評価する
4) 最大確率の文が元の語順を復元できたかどうかを評価する

Brown 等は長さ 11 単語以下の 38 文を対象に 3−gram を使った実験を報告しており、これによると 63%（24 文）の文を正確に復元できたとしている。さらに意味的に正しい文まで正解にすると 84%（32 文）の正解率であったと報告している[1]。

この実験を受けて、1994 年に日本 IBM の丸山が日本経済新聞 1 年分のデータを使った実験を報告している(丸山 94)。この実験は、英語と性質の違う日本語で、どの程度 N−gram が働くかを確認する目的で行っている。日本語は英語に比べて、

○ 単語の切れ目が明瞭でない、
○ 文法が語順に与える制約が緩やかである、
○ 一般に、語彙の大きさが大きい、

ことから英語ほど N−gram が働かないだろうという予想をしている。

しかし実験の結果、予想に反して Brown 等の英語の実験と同等の正解率を得たとしている。この実験では新聞 11 ヶ月のデータを N−gram の学習に使い、1 ヶ月のデータの中から長さ 10 単語の文を抽出して評価をしている。この結果、2−gram で 24%（394/1628）、3−gram で 39%（190/482）の正解を得たとしている。

著者等は講演などの独話を対象とした翻訳システムの実現を目指した研究を進めている。その実現手法としては、統計翻訳をはじめとするコーパスを用いたデータドリブン翻訳手法を検討しており（Tanaka 02 ）、これに利用するさまざまな言語データを収集している。今回、これらのデータを利用して”bag translation”の実験を実施した。目的は下記項目の調査である。

日経、NHK、フレーズブックなど性質の異なるコーパスでの正解率の差
日英の言語差による正解率の差
解の探索における枝刈りと実行時間の関係、枝刈りと正解率の関係

---

[1] 実験に使ったデータベース、学習と評価に使ったデータの量など詳細は不明である。

本報告の構成は以下の通りである。まず2章で実験の概要を説明し、3章では実験に利用した言語資源と実験の基礎になるN−gram作成について説明する。続く4章では基本的な実験、文長の影響、探索時における枝刈りの影響を調査した実験を報告し、5章で本報告のまとめを行う。

2 実験概要

実験では、以下の図に示されるように、NHK ニュースデータ（日英)、フレーズブックの収集データ（日英)、日経新聞（日本語のみ）のコーパス（すべて茶筅による形態素解析済み）それぞれに対して 90％を学習データとして、CMU の N グラム作成ツールキット（CMU-Cambridge Statistical Language Modeling Toolkit v2）を利用して N グラムモデルを作成した。また、これらのコーパスそれぞれに対する 10％をテスト用データとして残し、その中から文長が 10 である文、1000 文（フレーズブックのオープンテストでは 1000 文抽出できなかった）をランダムに選び出し、それらをオープンのテスト文とした。長さを 10 と固定したのは、実験を実用的な時間内に終わらせるためである。また、単語の並べ替え時のサーチは、分枝限定法（branch and bound(矢野 86)）に従った。

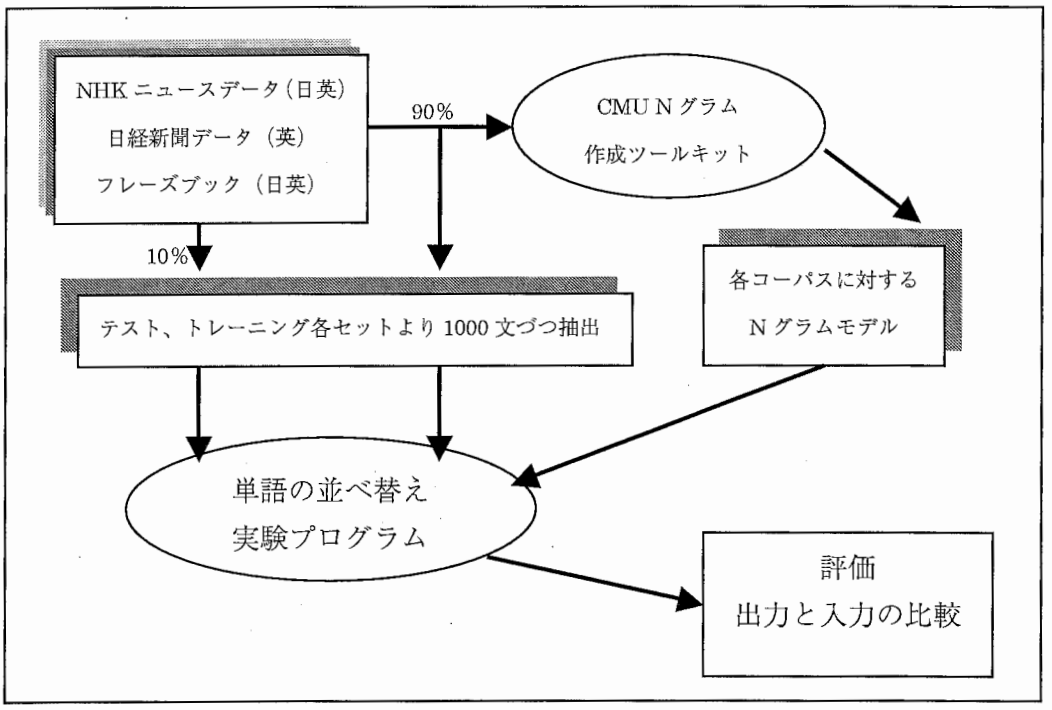

図 1 実験概要

3 言語資源とN−gram作成

3.1 言語資源

先に述べたように、実験では、NHKニュース、日経新聞、フレーズブックの3種類のコーパスを用いた。日経新聞が英語のみ、他は日英両方の実験を行った。以下に、実験に使った言語データのコーパスサイズを示すが、日経新聞の日本語のコーパスは非常に大きく、CMUのNグラムツールキットでのNグラムデータ作成に失敗したので実験は行わなかった。NHKニュースは1995年4月から2000年3月まで、日経新聞は、1995年1月から2000年12月までのデータとなっている。フレーズブックは、146,105対の複数文どうしの対応も含む日英対応になっている。

コーパス名称	言語	記事 (K)	文 (K)	単語 (K)	単語／文
NHKニュース (95/04-00/03)	日	287	1593	75602	47.45
	英	74	559	12252	21.95
日経新聞 (95/01-00/12)	日	1758	18620	499405	26.82
	英	184	1416	31909	22.53
フレーズブック	日		172	1011	5.86
	英		172	1185	6.87

表 1 実験に利用した各コーパスの大きさ

3.2 Nグラムモデルの作成

前述のデータを形態素解析ツール茶筅により解析したもの（分かち書きにし、読点、鍵括弧など余分な記号は取り除いた）を入力にしてNグラムモデルを作成した。Nグラムモデルの作成には、CMUのNグラム作成ツールキット（CMU-Cambridge Statistical Language Modeling Toolkit v2）を使った。以下にその作成手順を示す。

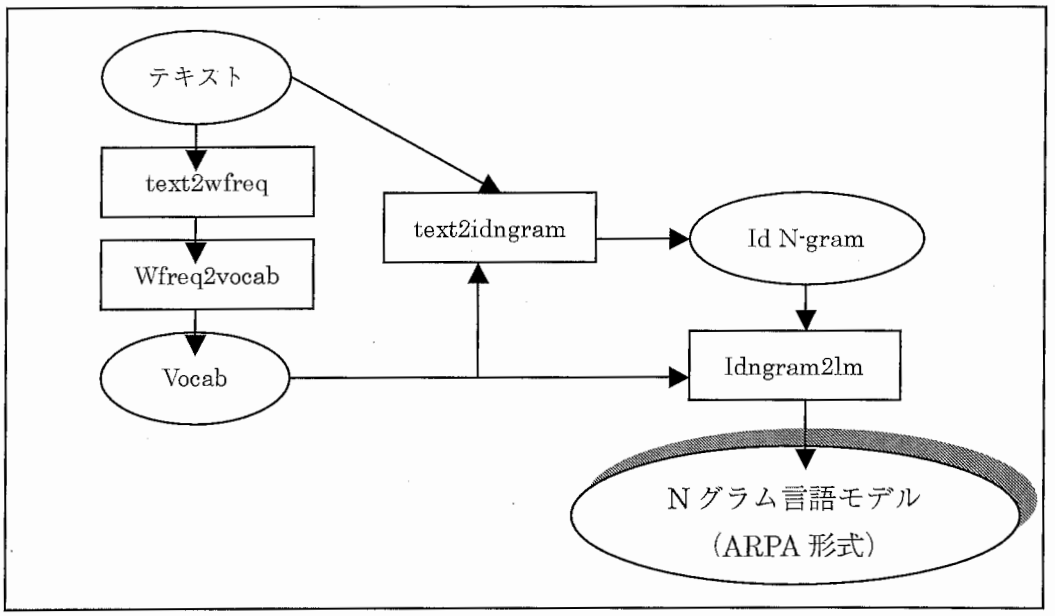

図 2　N-gram 作成手順

具体的には次のようなコマンドで作成する。

```
(例) Nグラムモデルの作成
# cat テキスト名 | text2wfreq | wfreq2vocab -top 20000 > XXX.vocab
# cat テキスト名 | text2idngram -vocab nikkeij.vocab | idngram2lm -vocab
  XXX.vocab -idngram - -arpa XXX.arpa -spec_num 5000000 15000000
```

N グラムモデルは、ARPA 形式のテキストファイルで作成した。ARPA 形式のテキストファイルは、以下のような形式を持つ。ARPA 形式の詳細については、他の資料を参照して頂きたい。

```
¥data¥
ngram 1=n1
ngram 2=n2
...
ngram N=nN
¥1-grams:
p w [bow]
...
¥2-grams:
p w1 w2 [bow]
...
¥N-grams:
p w1 ... wN
...
¥end¥
```

Nグラムモデルを作成する際の語彙の上限は、20K語とした。以下に示すように、20Kの語彙サイズで日経、NHK日英コーパス中のトークンの96%から98%をカバーする。

コーパス名称	言語	カバー率
NHKニュース	日	98.4%
(95/04-00/03)	英	97.9%
日経新聞	日	96.4%
(95/01-00/12)	英	97.3%

表２２０K語彙でのコーパス単語のカバー率

以下に、作成されたNグラムモデルのデータの異なり数を示す。

コーパス名称	言語	1-gram	2-gram	3-gram
NHKニュース	日	20,001	1,725,906	9,303,197
(95/04-00/03)	英	20,001	1,133,193	4,172,670
日経新聞	日			
(95/01-00/12)	英	20,001	2,105,371	9,185,766
フレーズブック	日	17,582	110,867	277,386
	英	11,253	97,452	263,172

表3　N-gramの異なり数

4　実験

4.1 探索アルゴリズム

日英単語の並べ替え実験においては、単語の長さ10の文を対象としている。すなわち、文の単語の並びの順列である10！（3,628,800）通りの候補が存在することになる。この中からNグラムモデルを用いて最大の確率の候補を選ぶためには、効率の良い探索手法が必要となる。今回の実験では、基本的には、今まで見つかった解のうち、確率最大のもので、branch-and-boundを行う深さ優先探索を採用した。この手法は全解探索である。

この手法では、A*サーチのようにスタックの中から常に確率最大の候補を取り出し、枝を伸ばしてはまたスタックに入れる。これを繰り返しながら、解が一つ見つかり、探索途中の解よりも確率が高ければサーチを終了する。そうでなければ探索を続行する。また、この手法を採用するだけでは、横に探索空間が広がる可能性もあるため、ビタビサーチのように状態を記録して、任意の状態における確率最大の候補のみを記録しながら枝を伸ばしていく方法を併用した。3グラムモデルでは、文頭からの長さと現在の単語と過去の単語2つの組み合わせ（wd1,wd2,wd3）を状態として持ち、同じ状態の候補は今までにサーチで見つかった確率最大の候補を一つ持つだけである。これらのサーチを実現するために一つの候補の情報を以下に示すようなpath構造に記録する。

```
path構造
  word              ;; 選択された最新の単語(wd3)
  rest-words        ;; 未選択の単語
  nth-order-words   ;; 過去の状態を表す単語列(wd1,wd2,wd3)
  position          ;; 文頭からの長さ、あるいは位置
  prob              ;; 部分確率、すなわちp(wd3|wd1,wd2)
  prob-so-far       ;; 文頭からの最適部分経路確率
  previous          ;; 直前の path 構造へのリンク
```

path構造は、サーチ途中での1つの候補の記録である。これらの候補は、path-listの中にスタックされる。サーチは、path-listの中からprob-so-farスロットに最大の確率を持つpath構造を取り出して、スロットにある単語を後に続く単語として新しいpath構造を作成して登録すると言う繰り返しで行われる。登録の際は、同じ状態の候補は確率最

大のもの１つのみが登録される。

branch-and-bound による深さ優先探索のアルゴリズム

```
Function search-wd-ngram-with-branch-and-bound (words)
begin
;; 初期パスリストの作成（先頭候補の作成と登録）
make-initial-path-list (words);
LOOP:
  if path-list == nil then exit(fail);
  ;; スタックから候補を取り出す。
  bestpath := pop(path-list);
  ;; 文末の確認と終了
  if bestpath.position == length(words)+1  then
    exit(bestpath.previous);
  ;; 文末の処理、句点を付けて最終確率を割り当てる。
  elseif path-position(bestpath) == length(words)  then
    newpath := make-path();
    newpath.word := DUMMY_SYMBOL;
    newpath.nth-order-words :=
          append(rest(bestpath.nth-order-words),list(DUMMY_SYMBOL));
    newpath.position := bestpath.position + 1;
    newpath.prob := get-n-gram-prob (newpath.nth-order-words);
    newpath.prob-so-far := bestpath.prob + newpath.prob;
    newpath.previous := bestpath;
    register-path-to-path-list(newpath);
;; 新しい候補を作成して登録する。
  else
    foreach word in bestpath.rest-words do
        newpath := make-path();
        newpath.word := word;
        newpath.rest-words := remove-words (word, bestpath.rest-words);
        newpath.nth-order-words :=
              append(rest(bestpath.nth-order-words),list(word));
        newpath.position := bestpath.position + 1;
        newpath.prob := get-n-gram-prob (newpath.nth-order-words);
        newpath.prob-so-far := bestpath.prob + newpath.prob;
        newpath.previous := bestpath;
        register-path-to-path-list(newpath);
    end
  endif
goto LOOP;
```

前述のように、基本的には、path-list というスタックへの候補の出し入れによりサーチを進める。path-list は、常に確率の大きい順にソートされている。もし、スタックから取り出した確率最大の候補（bestpath）が文末の処理が済んだものであれば、それを解として出力する。しかし、文末の処理が済んでおらず、文末に達している場合、後続に DUMMY_SYMBOL が続く確率、すなわち文末になる確率を計算し、それをその候補に加味して、再びスタックに入れる。初期の path-list は、次ページに示すように、各単語が DUMMY_SYMBOL に続く確率、すなわち、先頭になる確率を計算して登録する（関数 make-initial-path-list)。文の長さが１０の入力では、それぞれを先頭の単語とする１０個の候補が登録される。DUMMY_SYMBOL は、日本語の場合、句点である。

path-list への登録の際は、登録する候補（path 構造）の前の単語２つと現在の単語（nth-order-words スロット）及び先頭からの単語の長さ（position スロット）をキーにし

て確率最大の候補のみを登録して、path-list 全体をソートする（関数 register-path-to-path-list)。

```
;; 初期パスリストの作成
Function make-initial-path-list (words)
begin
path-list := nil;
foreach word in words do
      newpath := make-path();
      newpath.word := word;
      newpath.rest-words := remove-words (word, words);
      newpath.nth-order-words := list(DUMMY_SYMBOL, DUMMY_SYMBOL, word);
      newpath.position := 1
      newpath.prob := get-n-gram-prob (newpath.nth-order-words);
      newpath.prob-so-far := newpath.prob;
      newpath.previous := nil;
      register-path-to-path-list(newpath);
end
```

```
;; 作成したパスの登録
Function register-path-to-path-list (newpath)
begin
key := list(newpath.position, newpath.nth-order-words);
oldpath := find-path(key ,path-list);
if oldpath == nil || newpath.prob-so-far > oldpath.prob-so-far  then
  pathlist := delete-path(oldpath, path-list);
  insert-and-sort-path(newpath, path-list);
endif
end
```

```
p(wd3|wd1,wd2)=
        if(trigram exists)  then
          p_3(wd1,wd2,wd3);
        else if(bigram w1,w2 exists)  then
          bo_wt_2(w1,w2)*p(wd3|wd2);
        else
          p(wd3|w2);
p(wd2|wd1)=
        if(bigram exists) then
          p_2(wd1,wd2);
        else
          bo_wt_1(wd1)*p_1(wd2);
```

また、関数 get-n-gram-prob における部分確率（ある単語列、wd1, wd2 の後に wd3 が出現する 3-gram モデルでの出現確率）は左に示すようにして求めている（実際の計算は確率値を対数に直している）。bo_wt_* は、back-off weights である。

その他、この実験で用いるデータがフレーズブックの収集データのように N グラムの異なり数がある程度小さい時には、すべての N グラムモデルをオンメモリで利用することが出来るが、NHK ニュースや日経のデータのような大きなデータでは、すべての N グラムモデルを常駐させることが不可能になる。そこで、今回の実験では、NHK ニュースや日経のデータを使う場合、1-gram,2-gram については、常にオンメモリで利用したが、3-gram については、一部を二次記憶に置くような措置を取った。当然、二次記憶を使う場合は処理時間が大きく増加した。しかしながら、実験としては現実的な時間内で処理を終えることができた。これらのインプリメントに関する詳細な説明はここでは行わない。代わりに、二次記憶を使う場合の簡単なイメージを以下に示す。

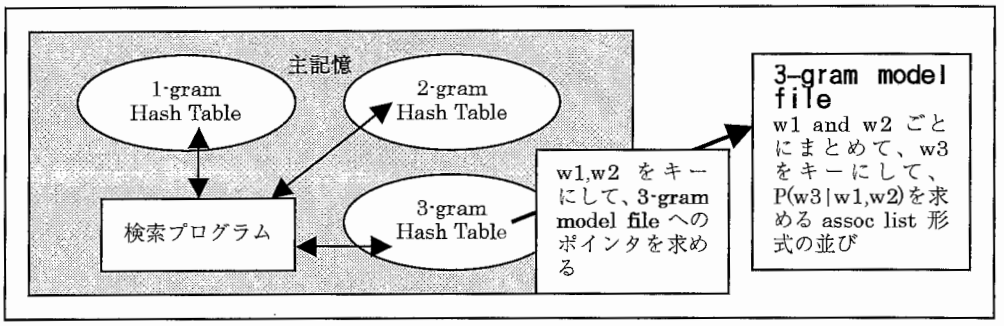

図 3　オフメモリモードのイメージ

4.2 基本実験と考察

このようにして得られた実験結果を以下に示す。ここでの評価は、入力と出力を比較して完全に一致した正解率であり、同じ意味で出力されても入力と順序の違うものは正解の対象になっていない。出力を見る限り、3 グラムモデルを使うと、順序は違うが、同じ意味で出力されていることが多く、また意味の一致とまで行かずとも統語的にはほぼ正しい文が出力されている場合も多い(16 ページ参照)。

コーパス名称	言語	2-gram		3-gram	
		open	closed	open	closed
NHK ニュース	日	11.8%(118/1000)	12.0%(120/1000)	33.4%(334/1000)	60.3%(603/1000)
(95/04-00/03)	英	15.6%(156/1000)	16.3%(163/1000)	25.2%(252/1000)	68.9%(689/1000)
日経新聞	日	24%	41%	39%	64%
(95/01-00/12)	英	14.0%(140/1000)	14.9%(149/1000)	26.2%(262/1000)	58.0%(580/1000)
フレーズ	日	22.7%(165/728)	36.6%(366/1000)	42.3%(308/728)	81.6%(816/1000)
ブック	英	19.8%(94/474)	22.6%(226/1000)	37.8%(179/474)	74.3%(743/1000)

表 4 日英 bag translation 正解率

先に述べたように、オープンテストは、10%のテストセットから、クローズドテストは90%のトレーニングセットの中から句読点、記号等を落として10語の長さの文1000文を対象に行った。フレーズブックは、テストセットの中に文長10の文が1000文に満たなかった（日本語728、英語474文）。

表中、日経新聞の日本語の正解率は、日本IBMの丸山らが1992年12月から1993年11月までのデータで同じ“bag translation”の実験を行った時の値である（丸山 1994）。

学習量

学習データ（close）と評価データ（open）の正解率の差は過学習の程度を示し、学習データ量の過不足をある程度評価する指標となる。

2-gramではフレーズブック日本語を除いて両者がほぼ同じ値であることから、フレーズブック日本語をのぞくと学習データ量の極端な不足はなかったものと考えられる。フレーズブック日本語だけなぜデータ不足が生じたかは現時点では不明である。

3-gramはすべてのデータで評価データでの正解率が学習データの正解率にくらべて小さくなっており、学習データ量がまだ不足していることを示唆している。

日英の言語差

（丸山 94）の研究の動機となった「日本語では英語に比べてN-gramがうまく働かない」という予想は我々の実験でも否定された。評価データの結果によるとNHKの2-gramを除いてむしろ日本語の正解率が高い結果となった。我々の実験は日英で同じ内容のデータを使った実験であり丸山の予想はかなりの確度で否定されたと考える。むろん日本語形態素単位の取り方による違い、純粋な話しことばでの実験など今後さらに調査を進めなくては、完全に結論づけることはできない。

日経とNHKの差

丸山による日経日本語の正解率と我々のNHKの日本語正解率にかなりの差がある。特に2-gramでの差が大きい。この原因はニュースデータの違い、形態素解析の違いが考えられるが、今回の実験の範囲では何ともいえない。丸山らはIBM独自の形態素解析器を使用し

ているため茶筅と形態素認定基準がかなり違っている可能性があり、この違いが大きいのではと考えている。今後、日経日本語の実験を行って正確な原因を突き止めたい。

フレーズブック

フレーズブックは日経、NHK に比べると高い正解率となった。ニュースにくらべて語彙が限られており、言い回しが限られていることが原因であると考えられる。先に述べたようにフレーズブックには 10 単語（形態素）の文は少なく、平均は日本語で 6 単語、英語で 7 単語程度である。このため平均的にはさらに高い正解率が得られる。この点は次節で議論する。

意味的、あるいは統語的に正しいと思われる例（NHK 日本語オープン）

正解：("更に" "二" "点" "差" "に" "迫ら" "れ" "た" "五" "回")

出力：("五" "回" "更に" "二" "点" "差" "に" "迫ら" "れ" "た")

正解：("横浜" "は" "立ち上がり" "五" "本" "の" "ヒット" "で" "四" "点")

出力：("横浜" "は" "ヒット" "で" "五" "点" "四" "本" "の" "立ち上がり")

正解：("続く" "三" "回" "渡辺" "秀一" "は" "コントロール" "に" "苦しみ" "ます")

出力：("渡辺" "は" "続く" "三" "回" "秀一" "コントロール" "に" "苦しみ" "ます")

正解：("その" "丸山" "十" "番" "ロング" "ホール" "の" "第" "三" "打")

出力：("その" "丸山" "の" "十" "番" "ロング" "ホール" "第" "三" "打")

正解：("足" "を" "生かし" "て" "内野" "安打" "塁" "に" "出" "ます")

出力：("内野" "安打" "塁" "に" "出" "て" "足" "を" "生かし" "ます")

正解：("フィス" "が" "二" "年" "連続" "三" "回" "目" "の" "優勝")

出力：("二" "回" "目" "優勝" "の" "フィス" "が" "三" "年" "連続")

正解：("ロッテ" "は" "オープン" "戦" "負け" "なし" "の" "三" "連勝" "です")

出力：("三" "連勝" "の" "ロッテ" "は" "オープン" "戦" "負け" "なし" "です")

正解：("打撃" "陣" "で" "は" "二" "年" "目" "の" "二" "岡")

出力：("で" "は" "打撃" "陣" "二" "年" "目" "の" "二" "岡")

正解：("一方" "サッカー" "の" "Ｊリーグ" "Ｊ" "１" "は" "八" "試合" "です")

出力：("一方" "Ｊリーグ" "は" "サッカー" "Ｊ" "１" "の" "八" "試合" "です")

正解：("高校生" "ルーキー" "河内" "は" "かわ" "うち" "きょう" "が" "デビュー" "戦")

出力：("河内" "かわ" "うち" "は" "高校生" "ルーキー" "きょう" "が" "デビュー" "戦")

正解：("日本" "勢" "で" "は" "尾崎" "直道" "選手" "が" "四十" "位")

出力：("選手" "が" "日本" "勢" "は" "四十" "位" "で" "尾崎" "直道")

正解：("カンボジア" "で" "電気" "泥棒" "が" "逮捕" "さ" "れ" "まし" "た")

出力：("カンボジア" "で" "逮捕" "さ" "れ" "まし" "た" "が" "電気" "泥棒")

正解：("建物" "など" "に" "被害" "は" "あり" "ませ" "ん" "でし" "た")

出力：("建物" "に" "被害" "など" "は" "あり" "ませ" "ん" "でし" "た")

正解：("その" "六" "回" "は" "最後" "の" "バッターペタジーニ" "を" "内野" "フライ")

出力：("その" "バッターペタジーニ" "は" "最後" "の" "六" "回" "を" "内野" "フライ")

4.3 文の長さの影響

フレーズブックデータ、NHKのニュースデータに対して、文に含む単語数を５から１１まで変えて、その正解率、処理時間がどのように変化するのか調査した。前回と同様、コーパスの９０％を学習データとして用いて3-gramの言語モデルを作成し（最初の実験で作成したものを利用)、残りの１０％の中からそれぞれの長さのものをランダムに１０００文（１０００文ないものは、取り出せただけ）取り出し、それらをテスト文として、"bag translation" 実験を行った。正解率は、入力との完全一致であり意味的に正しいものが出力されても、もとの並びと違えば誤りと判断される。

なお、平均処理時間はテスト文１文に対する平均処理時間であり、全文の処理時間（実時間には実験状況によって揺れが生じたため、Allegro Common LispのTime関数によって計測したTotal Run Time（CpuTime）を採用した）をテスト文数で割ったものである。

表５および図４に文長と処理時間の関係を示す。表６および図５に文長に対する正解率の推移を示す。

文長	最大組み合わせ数	PB(日)	PB(英)	NHK(日)	NHK(英)
5	120	3	3	475	341
6	720	13	11	1,480	756
7	5,040	76	59	3,919	1,489
8	40,320	437	353	9,852	4,202
9	362,880	1,863	2,322	23,277	9,488
10	3,628,800	16,011	15,679	54,071	35,105
11	39,916,800	131,754	117,563	168,334	114,385

表 5 文長と平均処理時間 (m sec)

文長	PB(日)		PB(英)		NHK(日)		NHK(英)	
5	82.0%	820/1000	74.2%	742/1000	77.1%	571/741	58.4%	97/166
6	75.9%	759/1000	67.1%	671/1000	58.6%	397/677	46.5%	147/316
7	69.8%	698/1000	63.0%	630/1000	54.3%	480/884	40.7%	208/511
8	58.9%	589/1000	53.4%	534/1000	47.2%	471/997	35.2%	268/761
9	52.7%	527/1000	41.3%	296/716	45.3%	453/1000	31.7%	313/988
10	42.3%	308/728	37.8%	179/474	33.4%	334/1000	25.2%	252/1000
11	30.3%	144/476	24.5%	64/261	30.4%	304/1000	17.2%	172/1000

表 6 文長と正解率 (正解数／データ数)

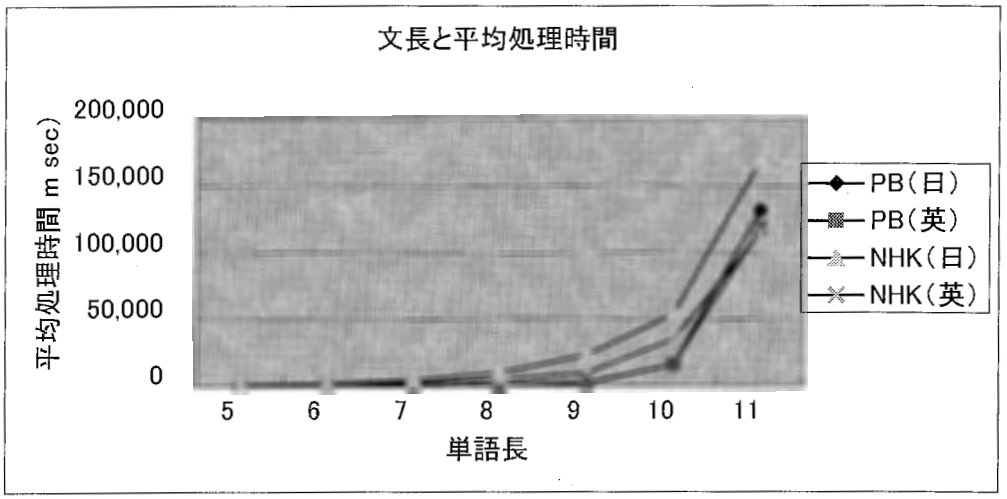

図 4　文長と平均処理時間

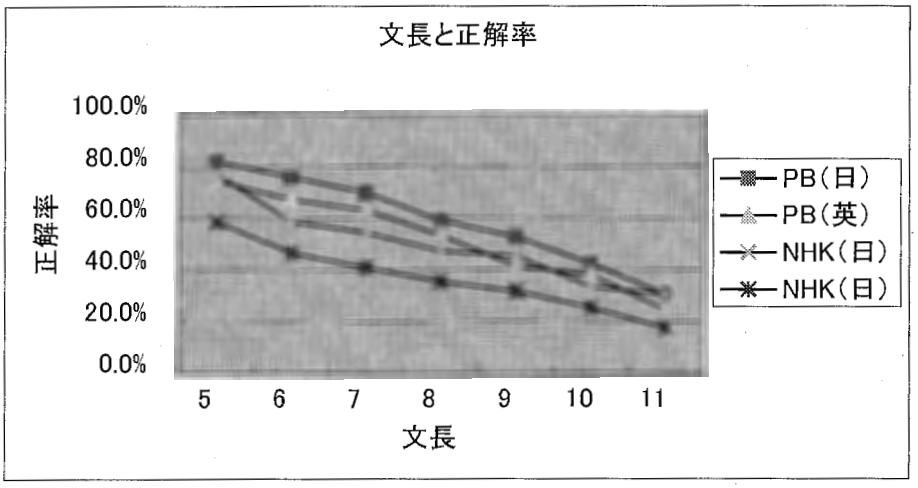

図 5　文長と正解率

4.3.1 文長と正解率

文長が長くなると正解率は直線的（線形的）に減少した。組み合わせ数は階乗のオーダーで増えるため偶然語順が回復される確率は指数的に減少する。これに対して正解率の減少が指数的でなく線形的にとどまったのは3−gramの効果といえる。単純な情報ではあるがその有効性は大きい。

絶対的な正解率については下記の点を指摘しておく。表1によるとフレーズブックの日本語の平均文長（形態素数）は6程度、英語は7程度である。この時の正解率は日本語で76%、英語で63%となる。この数字は今回の手法の単純さ、評価の厳しさ[2]を考えるとかなり高く、"bag translation"を何らかの処理に応用することを検討できるレベルだと考える。

一方NHKの平均日本語文長は47で、同英語は22である。このように長い文を対象に直接的な"bag translation"を行うことは精度の面で問題となる。それでも短い文では正解率が上がっている点は注目に値する。何らかの分割処理を行えば"bag translation"を有効に応用できる可能性はあると考える。

4.3.2 文長と処理時間

組み合わせ数を考えると、単純な枚挙法による処理時間は指数的な増加をする。今回の処理時間の増加も、図4に示すように処理時間は指数的であった。それでは分枝限定の効果はなかったのだろうか。これを確認するため組み合わせ数に対する処理時間の増加を調査した。

表7は、文長の増加に対する単語並び替えの組み合わせ数と処理時間の増加を比較したものである。フレーズブックの日本語の実験において、文長が5の時の最大組み合わせ数（5！）、平均処理時間を1とした時に、何倍になったかを示している。図6は表7のグラフである。

文長	最大組み合わせ数	平均処理時間
5	1.0	1.0
6	6.0	3.6
7	42.0	22.2
8	336.0	127.4
9	3,024.0	543.2
10	30,240.0	4,667.9
11	332,640.0	38,412.2

表 7 文長と最大組み合わせ数、平均処理時間比の関係

[2] 文単位の正解率であり、評価としては厳しい手法である。

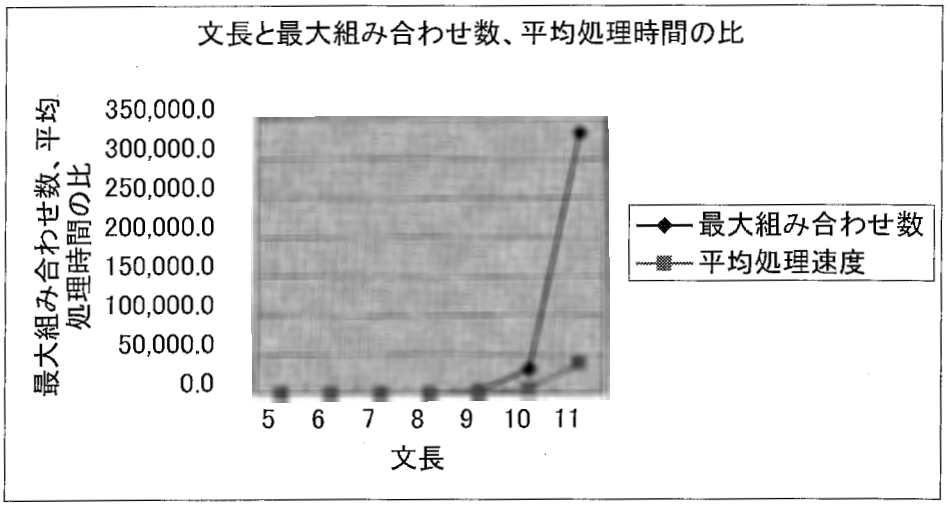

図 6　文長と最大組み合わせ数、処理時間比

これを見る限りでは、最大組み合わせ数の増加具合ほど処理時間は増加しておらず、分枝限定法がある程度効力を発揮したと結論できる。しかし、指数的な性質は変わっていないように見える。このため指数的な増加を防ぐための手法を検討調査する必要がある。この一つの手法が探索の枝刈りである。次節ではこの効果を検証する。

4.4 探索時の枝刈りの影響

文長を５単語から１１単語まで変えた実験で、文の長さが長くなれば、処理時間が急激に増加することが分かった。そこで、探索に使うスタックに大きさの制限を設けて枝刈りを行う実験をした。この時に正解率がどの程度低下し、一方、処理時間をどの程度短縮できるのかを調査した。前述の実験は、branch-and-bound のアルゴリズムを用いて全解探索を行うものであったが、ここでは、途中で膨らむ候補の格納数の制限を設け、それらの制限数を変動させた。具体的には解を保持するスタックサイズを変更した。

フレーズブックの収集データ、NHK のニュースデータに対して、スタックサイズを５０から２００まで変動させ、その正解率、処理時間について調べた。前述の実験と同様、コーパスの９０％を学習データとして用いて 3-gram の言語モデルを作成し（当初の実験で作成したものを利用)、残りの１０％の中からそれぞれの長さのもの１０００文を取り出し（１０００文に満たない時は取り出せただけ)、それらをテスト文として、"bag translation" 実験を行った。正解率は、入力との完全一致であり、統語的、意味的に正しいものが出力されても、誤りと判断されるのは前述の実験と同じである。

ここで言うスタックサイズは、全候補を制御する前述の path-list の最大サイズであり、この大きさを超える解候補が出現した場合、確率の低い候補から捨てた。今回は通常のビタビサーチではなく、branch-and-bound のアルゴリズムを採用していることから、探索段階の各状態における候補を格納するスタック（例えば、文頭からの長さが同じ候補の集合）は特別に設けていないので、これを使った枝刈りは行わなかった（今回のアルゴリズムでは枝刈りするのにコストがかかるため)。

次ページの表８、９および図７、８は実験結果をまとめたものである。表９および図８の平均処理時間は 4.3 節の実験と同様にテスト文１文に対する平均処理時間であり、全文の処理時間をテスト文数で割ったものである。また、表８、図７の一致率はスタックサイズ制限なしの結果との一致率である。一致率が高くても、スタック制限無の結果との一致がたかいだけで正解とは限らないことに注意されたい。

	PB(日)		PB(英)		NHK(日)		NHK(英)	
スタックサイズ	正解率	一致率	正解率	一致率	正解率	一致率	正解率	一致率
50	39.6%	64.7%	37.1%	74.7%	31.5%	73.3%	23.5%	63.5%
100	42.0%	79.0%	37.6%	85.4%	33.0%	89.8%	25.0%	82.5%
150	42.4%	84.9%	37.8%	88.8%	33.2%	95.5%	25.1%	91.3%
200	42.3%	88.3%	37.8%	91.1%	33.3%	97.5%	25.1%	94.6%
制限なし	42.3%	100.0%	37.8%	100.0%	33.4%	100.0%	25.2%	100.0%

表 8 スタックサイズと正解率、スタック制限無との一致率

スタックサイズ	PB(日)	PB(英)	NHK(日)	NHK(英)
50	225	156	24,990	9,640
100	714	501	35,905	14,190
150	1,433	985	42,956	17,574
200	2,230	1,580	45,692	20,322
制限なし	16,011	15,679	54,071	35,105

表 9 スタックサイズと平均処理時間（msec）

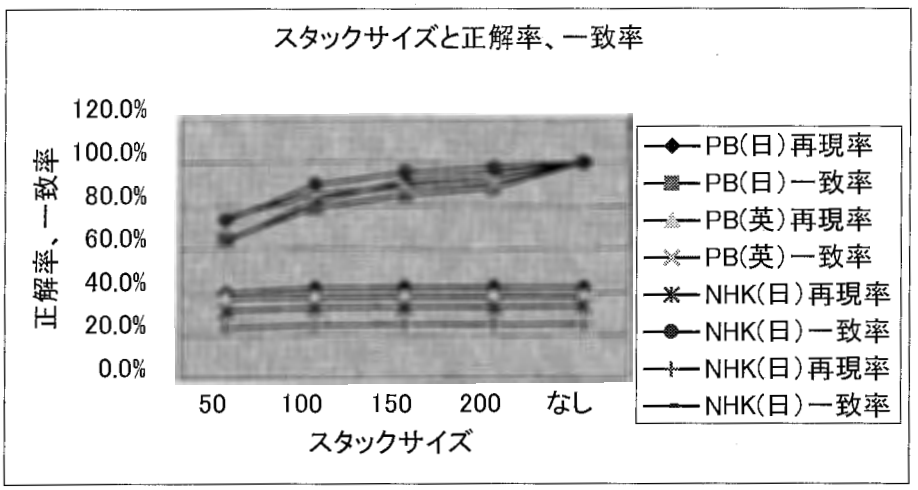

図 7　スタックサイズと正解率、一致率

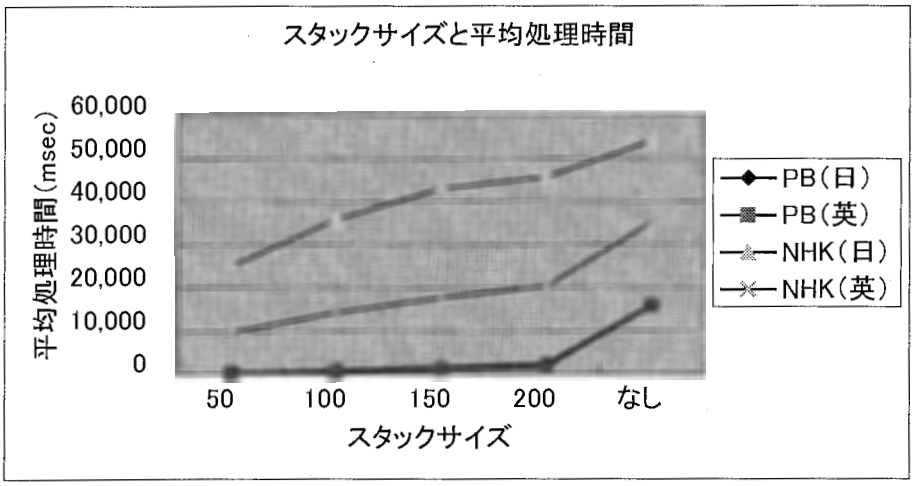

図 8　スタックサイズと平均処理時間

4.4.1 スタックサイズと正解率

フレーズブックの日本語の結果を見ると、スタックサイズ制限なしで branch-and-bound を行った時、各文における探索時の最大候補数（最大スタックサイズ）は、最大で 2074、最小で 46、平均 701.9 であった。このような数字を念頭においてスタックサイズを変化させた。

表 8 に示すように、スタックサイズ 100 でスタックサイズ制限なしとほぼ同じ正解率が得られ、スタックサイズ 200 以上（250、300 でも同じ結果が得られた）では、全く同じ正解率が得られた。

しかし、スタックサイズ制限無との結果の一致率は、スタックサイズ 200 でも 88.3%に留まった（ちなみに、スタックサイズ 250 で、90.5%、スタックサイズ 300 で 94.2%という結果が得られた)。すなわち、スタックサイズを大きくするとスタックサイズ制限無の解と一致する度合いは増加したが、正解率は増加しなかったことになる。フレーズブックの英語、NHK のニュースデータ（日、英）でも同様の傾向が見られる。

4.4.2 スタックサイズと処理時間

フレーズブック日本語に対する実験の結果では、スタックサイズが 50 から 200 へと 4 倍になると、処理時間は約 10 倍に増加した。またスタックサイズを制限無にすると処理時間は 70 倍に増加した。このような傾向はフレーズブックの英語、NHK のニュースデータ（日、英）においても見られた。

以上のことから、スタックサイズを増加しても、正解にたどり着く可能性は頭打ちとなり、ある程度以上のスタックサイズは単に探索時間を無駄にしていると結論できる。今回の実験によると、スタックサイズ 50 程度で全解探索とほぼ同じ効果がえられ、またその時にはかなりの高速化が実現できることが分かった。枝刈りは有効な手段であるといえる。

5 おわりに

NHKニュース日英データ、日経日本語ニュースデータ、フレーズブック日英データを対象とした"bag translation"の実験を行った。本実験で得られた主要な結論は下記の通りである。

○ データ量

学習データ、評価データでの正解率を観察した結果、データの量は2−gramを作成するにはほぼ十分であるが、3−gramを作成するには不十分であることがわかった

○ 日英の言語の差

どちらかの言語の方が容易であることを示す結果は得られなかった。特に、言語モデルは日本語ではうまく働かないという予想(丸山 94)は当たらず、むしろ日本語の方が好成績であった

○ 文長と処理時間

文長増加に対する処理時間の増加を調査した結果、分枝限定法による高速化の効果は確認できた。ただし文長が長くなると現実的な時間での探索は不可能となる。現在の計算機環境では11単語の並べ替えの完全探索を2ないし3分で実行できる状況である

○ 文長と正解率

文長が長くなるほど正解率は下がったがその減少度合いは指数的ではなく線形的であった。すなわち3−gramの効果は確認できた。またフレーズブックについては平均的な文長での正解率が日本語で76%、63%に達した。

○ 枝刈り

探索範囲を制限する実験では全体候補を50程度にしても全解探索とほぼ同程度の正解率を達成できた。またこれによって処理を70倍に高速化できた。

著者等は最終的には「あすを読む」のような講演調の音声言語、すなわち独話を対象とした翻訳手法を開発したいと考えている。その一つの候補として統計を駆使したデータドリブン翻訳手法を考えている。すでにフレーズブックのように語彙が制限されており、かつ文が短い音声言語であればIBM流の統計翻訳である程度翻訳できることが示されつつある。

本実験ではフレーズブックとの比較も行っており、統計手法の有効性をある程度予測できるのではと考えている。今後はこの実験結果を子細に検討して翻訳システム提案へとつなげて行きたい。

参考文献

Brown 90: A Statistical Approach to Machine Translation, Brown, P. F., J. Cocke. S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer and P. S. Roossin, Computational Linguistics, Vol. 16, No.2, , pp.79-85, 1990

Tanaka 02: Speech to speech translation system for monologues- data driven approach, Tanaka, H., S. Nightingale, H. Kashioka, K. Matsumoto, M. Nishiwaki, T. Kumano, T. Maruyama, ICSLP 2002, 2002 (to appear)

丸山 94：N グラムモデルによる、日本語単語の並び換え実験，丸山 宏，情報処理学会第 49 回全国大会，3－181，1994

矢野 86：数学ハンドブック，矢野健太郎監修，森北出版，1986