6G-1

マイコン上での機械翻訳システムの試作

諸星　博司　(富士通)
田中　穂積，糸井　理人　(東工大　工学部)

1．はじめに

電子技術総合研究所で開発された融合方式による英日機械翻訳システム<1>は、DEC-20の STANDARD LISP上で動作している。この翻訳システムは、規模が小さいこともあって、辞書と文法を主記憶上に置いている。

しかし、このシステムをそのまま現在のマイコンにのせるためには、メモリ容量が十分に確保できないこともあって、辞書と文法を主記憶上に置くことができない。そのために、辞書と文法を補助記憶に置かなくてはならない。本稿では、この問題を解決するための技術的方策と実験結果について述べる。

なお、今回の試作の環境は以下の通りである。

ハードウェア：富士通 FM-11 EX,ﾊｰﾄﾞﾃﾞｨｽｸ
ＯＳ　：CP/M-86
ＬＩＳＰ　：LISP86ｲﾝﾀﾌﾟﾘﾀ(北大　山本)
ｾﾙ数=29k，ﾌﾙﾜｰﾄﾞｾﾙ数=3k
翻訳システム：21kｾﾙ　(辞書，文法除く)

2．辞書の構成法

辞書はハードディスクに置かれ、その形式はCP/M-86 の標準ファイル形式であり、単語はアルファベット順に並んでいる。標準ファイルは、シーケンシャルファイルであるが、ＯＳにより“レコード番号＋レコード内相対アドレス”（以後インデックスと呼ぶ）によってランダムアクセスが出来るようになっているため、これを利用して単語単位のアクセスが行なえる様にした。（なお、オリジナルのLISP86にはランダムアクセスの機能がないため、独自に関数の追加を行なった。）

この様なランダムアクセスを用いて、図１に示す単語とそのインデックスの対応リストを作成した。このリストは、単語を１文字づつに分解して木構造に構成された形であり、葉の部分にインデックッスが付けられている。このような構成法により、辞書に要するメモリ領域の節約をはかることができる。

しかし、この対応リストをそのまま主記憶上に置いたのでは大幅なメモリの節約にはならない。なぜなら単語数に比例して対応リストが増大し、これが主記憶を圧迫することになるからである。そこで、この対応リストもまた先頭文字のグループに分割して、図２の様なインデックスファイルをハードディスク上に置くことにした。そして、このインデックスファイルのインデックスリスト（マスタインデックス）を作成しこれを主記憶上に置くことにした。

```
(66 NIL (89 ((52 . 47)))
        (85 NIL (89 ((50 . 31)))
                (83 ((46 . 109))))
        (73 NIL (89 ((44 . 10)))
                (79 NIL (75 ((42 . 35))))
                (68 NIL (83 ((41 . 33)))))
        (69 ((33 . 109))
                (68 ((37 . 111)))
                (69 NIL (78 ((37 . 75)))))
        (65 NIL (84 NIL (72 ((32 . 81))))
                (80 NIL (73 NIL (67 ((31 . 85)))))))
(65 ((5 . 120))
        (84 ((31 . 59)))
        (82 NIL (77 ((28 . 96)))
                (69 ((28 . 59))))
        (78 ((18 . 42))
                (68 ((21 . 70))))
    "   (77 ((17 . 102))
                (79 NIL (78 NIL (71 ((18 . 9))))))
        (76 NIL (76 ((15 . 56))))
        (70 NIL (84 NIL (69 NIL (82 ((15 . 23)))))))
        (68 NIL (68 NIL (76 NIL (69 ((13 . 45)))))
                (74 ((11 . 45))))
        (67 NIL (73 NIL (68 ((7 . 110)))))
        (66 NIL (79 NIL (85 NIL (84 ((6 . 25)))))))
```

```
BY      BUY     BUS     BOY     BOOK    BODY    BED
BEEN    BE      BATH    BASIC

AT      ARM     ARE     AND     AN      AMONG   AM
ALL     AFTER   ADDLE   ADJ     ACID    ABOUT   A
```

図１　単語とインデックスの対応リスト
（リスト中の各文字は
アスキーコードで表わされている）

```
((65 (81 . 7))   (66 (75 . 22)) (67 (68 . 38))
 (68 (61 . 23))  (69 (55 . 95)) (70 (51 . 108))
        :
        :
 (86 (5 . 118))  (87 (1 . 45))  (89 (0 . 0)))
```

図２　マスタインデックス

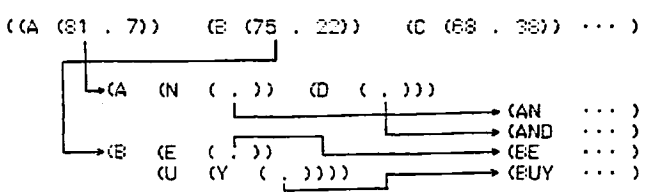

図３　辞書構成の概要

これにより、インデックスシーケンシャルファイル風なランダムアクセスを可能にした。図３に全体の概要を示す。

３．文法用ファイル

文法それ自体を記憶するために、文法規則数が増大すると多くのメモリを必要とする。さらに、本機械翻訳システムでは、文法から得られる予測情報を用いながら横型探索法によるパージングを行なっているため、多数の可能性を同時に保持しつつパージングを進める必要がある。そのため、メモリのオーバーフローは主にパージングの時に発生する。これを避けるために、文法を補助記憶に置くことが考えられる。

文法は、効率の点から見て主記憶上に置くことが望ましいのであるが、メモリ容量の立場から我々はこれをハードディスクに置くことにした。アクセス方法は辞書と同じランダムアクセスであるが効率を考えて、文法に対するインデックス全てを主記憶上に置くことにした。

次に、パージングにおいて読み込まれた文法は主記憶上に残さなくてはならないが、当然のことながら不必要な文法も数多く読み込まれることになる。そのため、読み込んだ文法を全て主記憶上に残したのでは空き領域を圧迫してしまう。そこで、メモリ節約のためにパージングに成功した文法のみを主記憶上に残し、他は主記憶上から削除することにした。

さらに、本機械翻訳システムでは、文法は図４に示す補強文脈自由文法を用いている。これは各文法規則に対して <sem>部が付いており、それにより意味処理に基づく翻訳処理を行なっているが、これをパージング中は対応するインデックスに置き換えておき、翻訳処理の時に改めて読み込む様にした。

以上の方策を採用した結果、パージングにおけるメモリオーバーフローの発生を抑えることができた。

```
(NT0 ((NT1 <advise 1>)
        :
      (NTj <advise j>))
     <cog>
     <sem>)
```

図４　補強文脈自由文法のLISP表現形式
(NT0 → NT1…NTj を表わす)

４．実験結果と考察

２と３で述べた方式を採用する前には、単語数を８３に制限した状態で６単語迄の単文しか翻訳できなかったが、本方式により８単語前後の単文，重文，複文迄の翻訳が可能になった。また、単語数はいくつでも追加することができる様になった。（現在は、２６７単語）

以下に入力文，翻訳結果，レスポンスをしめす。

```
I TAKE MY CHILD TO SCHOOL.
syn-time=30+28sec
私は学校へ私の子供を連れていく。
sem-time=32+43sec

I TAKE MY CHILD TO SCHOOL WITH HER.
syn-time=151+65sec
私は彼女といっしょに学校へ私の子供を連れていく。
sem-time=298+145sec

I TAKE A BUS AND PLANE.
syn-time=86+12sec
私はバスと飛行機に乗る。
sem-time=29+4sec

I HAVE ONCE PASSED  MOSCOW.
syn-time=79+11sec
私は１度モスクワを通過したことがある。
sem-time=32+4sec

THIS IS A FILM WHICH YOU DEVELOP.
syn-time=72+15sec
これはあなたが現像するフィルムである。
sem-time=37+14sec
```

レスポンスの悪い原因としては、ハードディスクへの多量のアクセスがあり、その対処が今後の問題である。

５．あとがき

本システムは、DEC-20のSTANDARD LISP 上で動作している翻訳システム全体をそのままマイコンに移植可能かどうかを調べること、また可能ならどの程度のパフォーマンスが得られるかを調べる目的で作成した。その結果、３２ｋセルでは翻訳システムとして限界が感じられる。しかし、６４ｋセル程度が利用可能になれば、相当長い文についても翻訳が可能であるという見通しを得た。

謝辞

本研究のために快くLISPインタプリタを提供してくださった北大・工学部・山本強氏に感謝致します。

参考文献

<1> 田中穂積：「解析から合成までを融合した英日機械翻訳システム」，日経エレクトロニクス，p.275-293 ，1983.8.29