技術報告

# MELCOM-COSMO 上での LISP 1.9 の開発†

畝　見　達　夫††

## 1. まえがき

LISP は人工知能や認知科学の研究を進める上で有用なプログラミング言語としてすでに一般的になっているにもかかわらず，わが国ではまだどの計算機上でも研究目的に使用できるほどには LISP の環境は整備されていない．

この度，開発した LISP 1.9 は電総研の TOSBAC-5600 上で動いていた LISP 2.0[4),6)]を MELCOM-COSMO-700Ⅲ上に移植するという形で導入したものである．電総研 LISP 2.0 は拡張 LINGOL[12)]など，幾つかの実績をもっており，TSS によるインタラクティブな使用に十分対応できるシステムである．

MELCOM-COSMO シリーズ計算機上で動く LISP としては，LISP 1.5[1),8)]及び Interlisp[3),9)]があるが，前者は機能の面で劣り，後者は極めて豊富な機能を持つ反面そのプログラムが重く，また両者とも処理速度が遅いことから筆者の研究目的には合致しないと判断される．

ハードウェアも OS もかなり異なる計算機間での移植作業であったために，データ表現などを根本的に変更する必要があった．また，単に機能をコピーするだけでなく，豊富なディスク I/O 機能の拡張や浮動小数点数演算の関数の追加を行った．

以下，2. 当システム開発の目的，3. 開発上の問題点，4. 当システムの特徴，5. 他システムとの比較，6. 今後の課題，について述べる．

## 2. 当システム開発の目的

人工知能及び認知科学研究の道具として実用に耐えうるものを作成することが当システム開発の目的である．当初の目標としては，電総研の LISP 2.0 上で動いている n 進木 LINGOL[13)]をのせることができる程度のものを目指したが，その後，第4章で述べるような機能の拡充を行った．

## 3. 開発上の問題点

開発に際して主な問題となったのはマシン及び OS の相違によるもので，1) 1ワード中のビット数，2) 入出力，3) 文字コード，である．

### 3.1 1ワード中のビット数とデータ表現

TOSBAC-5600 は1ワードが36ビット構成であることから，電総研 LISP 2.0 では1ワードを18ビットずつに分けて1ワード中に car 部と cdr 部を含む1セルを表現している．これに対し，COSMO-700Ⅲは1ワード当り32ビットなので，これを半分に分けて16ビットずつにすると，ポインタが指示できる範囲が64Kセルとなり大きなプログラムを動かすことが困難になってしまう．そこで当システムでは2ワードで1セルを表現することにした．

電総研 LISP などのデータ表現については他の文献[5)]に詳しいのでそちらを参照願いたい．当システムでは **図-1** のように左側（先頭）の1バイト（8ビット）に記録した目印によってデータ型を区別する方式を採用している．このため，1つのデータ型がメモリ上のあるアドレス範囲にまとまって存在する必要はなく，より柔軟なメモリ割付けが可能である．具体的にはゴミ集めの効率も考慮して，**図-2** に示すようにメモリの動的割り当て領域を7つに分割して使用している．SUBR 関数等として定義されているアトムのアトムヘッダはインタプリタ内に埋め込まれている．また，ポインタそのものでその値を表現する small integer の範囲が大幅に拡大するので，整数値計算におけるゴミの発生がかなり抑制される．その他，アトムヘッダ，ストリングヘッダ及び配列の構造も変更されている[14)]．

### 3.2 入　出　力

当システムでは TSS を用いたインタラクティブな

† Development of LISP 1.9 on MELCOM COSMO by Tatsuo UNEMI (Dept. of System Science, Graduate School of Science & Engineering at Nagatsuta, Tokyo Institute of Technology).

†† 東京工業大学総合理工学研究科システム科学専攻

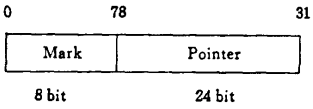


図-1 1ワード中のデータ表現

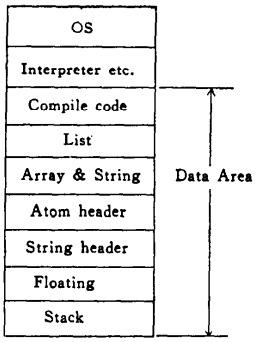


図-2 データ領域の構成

使用を前提としているので，入出力は端末と磁気ディスクに対してだけ可能である．

### 3.2.1 端末入出力

端末に対しては，文字単位及び行単位の入出力が可能である．行単位の入出力はメモリ中に確保されたバッファを通じて行われ，1行の入力が完了するまではプログラムは待ち状態となるが，文字単位の入力では，キーを押した時点で直ちにその ASCII コードがプログラム側に知らされる．

出力について特に問題となるのは改行を伴わない出力関数である．電総研 LISP では，これらの関数を実行しても出力されるべき文字列がメモリ中のバッファに貯えられるだけで，改行を伴う関数が実行された時点で初めて出力される．これに対し当システムでは改行を伴わない文字単位の出力が可能なので，メモリ中のバッファを介さずに関数の実行と同時に直接出力するようになっている．この機能は入力催促 (prompt) 文字列やグラフィック端末等の制御文字の出力に便利である．

### 3.2.2 ディスク入出力

ディスクファイルに対する入出力は行（レコード）単位でしか行えないので，READ, PRINT などの機能は電総研 LISP と同じである．ただし，バイナリ入出力やランダムアクセス機能については計算機システムの OS に依存するため，当システム特有のものになっている．詳しくは次章で述べる．

### 3.3 文字コード

ASCII コードを用いている電総研 LISP および EBCDIC コードを用いている MELCOM-COSMO-UTS/VS 上の他のシステム[10]と当システムとの互換性をはかるため，文字コードについて若干の工夫がなされている．

すなわち，LISP システム中の文字コードに関する関数の機能を電総研 LISP と同じにするため，メモリ中では ASCII コードを使用し，ディスクファイルについては，UTS/VS 上のテキストエディタや FORTRAN など，他の言語で書かれたプログラムとの互換性をはかるため EBCDIC コードを用いている．これを実現するため，READ, PRINT などによるディスクファイルアクセスの際に，読み込み時には EBCDIC から ASCII へ，書き込み時には ASCII から EBCDIC へのコード変換を行う．また，バイナリ入出力関数を用いて文字列の入ったファイルをアクセスする場合のことも考慮して，入出力関数とは別に，文字コードの変換のみを行う関数 ASC->EBC 及び EBC->ASC も用意している．コード変換の全体的な図式を 図-3 に示す．

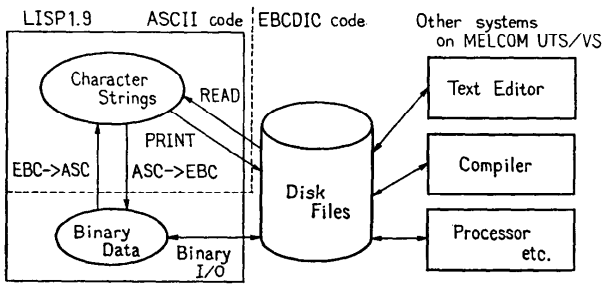


図-3 2種類の文字コードの使用

## 4. 当システムの特徴

機能的にはほぼ電総研 LISP 1.9 と同じなので，詳しくはそれに関する文献4)を参照されたい．電総研 LISP と異なる点では次のような特徴がある．

1. 磁気ディスク入出力機能を豊富に備えている．
2. 浮動小数点数の演算が FORTRAN タイプの算述式で行える．

### 4.1 磁気ディスク I/O

当システムでは UTS/VS 上の 1. コン

ゼクティヴファイル，2. キードファイル，3. ランダムファイル，という3種類の編成のディスクファイル[10]全てに対する入出力を用意して，あらゆる要求に対応できるようにしている．コンゼクティヴファイルはシーケンシャルにしかアクセスできないが，キードファイルとランダムファイルでは任意のレコードに対するアクセスが可能である．

バイナリ入出力及びランダムアクセスには，ともにSUBR 関数 KREAD, KWRITE を用いる．形式は次のとおりである．

(KREAD FL AR KY SZ)

(KWRITE FL AR KY SZ)

FL：ファイル名（アトム）．

AR：入出力バッファとなる配列（アトム）．

KY：ランダムアクセス時のレコード指定．

SZ：データの長さ（単位＝バイト）．

キードファイルの場合は第3引数KYに整数またはストリングを入れることによってキーを指定する．キーには任意長のバイト列を与えることができるので，データベースの構築に便利である．ランダムファイルの場合はレコード番号を指定する．

ランダムアクセス用の関数としては上述のバイナリ入出力用のものしか用意されていないが，S一式の入出力をランダムアクセスでも実現できるように，当システムでは，バッファとして使用される配列中のコード列をS一式に変換する関数 READARRAY 及び配列中に文字コード列を出力する関数 PRINTARRAY を用意している．

### 4.2 浮動小数点数の算述式演算

この機能は MLISP[2]や Interlisp の CLISP[3]と呼ばれる機能の中にも組み込まれているが，当システムではそれらとは若干異なり，FSUBR 関数 FCALC を用いて以下のように記述する．

(FCALC 〈算述式〉)

〈算述式〉::=〈項〉|〈算述式〉〈空白〉〈2項演算子〉〈空白〉〈項〉

〈項〉::=〈定数〉|〈変数〉|〈単項演算子〉〈空白〉〈項〉|(〈算述式〉)|@〈空白〉〈S一式〉

〈2項演算子〉::=＋|－|＊|％|^

〈単項演算子〉::=＋|－|ABS|SQRT|LOG|LOG10|ANTILOG|SIN|COS|TAN|ARCSIN|ARCCOS|ARCTAN|TANH

たとえば二次方程式の大きい方の根を求める式は次のように書くことができる．

(FCALC (－ B ＋ SQRT (B ＊ B － 4.0 ＊ A ＊ C)) %(2.0 ＊ A))

このような仕様にしたのは，当システムにこの機能を組み込んだ目的が，LISP 特有の算述式の冗長な表現を避けることではなく，浮動小数点数演算によって多くのゴミが出るのを避けることにあるからである．算述式中の演算の途中結果は LISP のデータ型の1つとして表現されずに CPU 内の汎用レジスタを介して次の演算に回される．一般的な LISP の表記法で書かれた浮動小数点数演算の式を評価すると，関数（演算子）の数と同じワード数のゴミが出るためゴミ集めが頻繁に起こるか，あるいは，それを避けるために余計なメ

表-1 他の LISP システムとの比較

| | | | LISP 1.9 on MELCOM | LISP 1.9 on TOSBAC | LISP 1.5 on MELCOM | Interlisp on MELCOM |
|---|---|---|---|---|---|---|
| 大きさ（ワード） | | | 13K | 18K | 8K | 88K |
| インタプリタ | TARAI | 3 | 1,101 | 271 | 2,543 | 2,405 |
| | | 4 | 21,055 | 5,124 | 51,322 | 45,468 |
| | | 5 | 572,587 | 138,819 | 1,427,994 | 1,236,566 |
| | SORT | 20 | 1,145 | 505 | 3,686 | 2,593 |
| | | 40 | 4,398 | 1,415 | 13,878 | 9,724 |
| | | 60 | 9,706 | 2,061 | 21,788 | 23,255 |
| コンパイラ | TARAI | 3 | 235 | 87 | — | 646 |
| | | 4 | 4,575 | 1,604 | — | 12,103 |
| | | 5 | 125,100 | 43,609 | — | 332,818 |
| | SORT | 20 | 126 | 31 | — | 609 |
| | | 40 | 447 | 108 | — | 2,359 |
| | | 60 | 987 | 176 | — | 5,257 |

unit＝ms.

モリを確保しなければならなくなる．このことはACT-E[11]に見られるような“強さ”を導入した認知科学モデルの LISP 上でのシミュレートに際して問題となる点であろう．

## 5. 他システムとの比較

大きさと処理速度について TOSBAC-5600 上の電総研 LISP 及び MELCOM-COSMO-700 Ⅲ 上の LISP 1.5 と Interlisp との比較を**表-1**に示す．当システム自体のプログラムはアセンブラで約 8 K ステップである．処理速度については第2回 LISP コンテスト[7]を参考にした．

大きさでは，機能を一部縮小した分だけ電総研 LISP 1.9 より小さくなっている．処理速度に関しては，同じ MELCOM-COSMO-700Ⅲ 上の他のシステムより2倍速くなっている．電総研 LISP 1.9 と比べると，4倍程度遅くなっているが，コンパイラを使用すれば n 進木 LINGOL などの実用には十分対応できる速度が得られる．

## 6. 今後の課題

LISP に限らず，プログラミング言語というものはその応用をもって評価されるべきであろう．現在当システム上で動いている応用プログラムとしては当初の目標でもあった n 進木 LINGOL のほか，本論文及び当システムのマニュアル作成のための簡単な日本語エディタや LISP 自身のコンパイラがある．今後，応用プログラムとして開発すべきものは，このようなプログラミング支援システムと当システム開発の目的である人工知能，認知科学研究用のプログラムである．

また，システム自体に関しては，電総研 LISP 2.0 に組み込まれている tuple, set などのデータ型[4]の導入について現在検討中である．

## 7. あとがき

もともと筆者自身 LISP 処理系については素人で，本来ならばあくまでもユーザの側にいるはずのところが，使えるものがなければユーザたりえないわけで，やむなく，ここで紹介したシステムを作成したというのが正直なところである．現在筆者は当システム自体のデバッグ及び機能拡張と応用プログラムの作成に取り組んでおり，これらが本領発揮である．当システムが幾らかでも人工知能や認知科学研究の発展に寄与できれば幸いである．

## 参考文献


1) McCarthy, J. 他: LISP 1.5 Programmer's Manual 2nd. ed., MIT Press (1965).
2) Smith, D. C.: MLISP, AIM-135 Stanford AI Project (1970).
3) Teitleman, W.: Interlisp Reference Manual, XEROX PARC (1974).
4) LISP 2.0 User's Manual, EPICS-5-ON-4, 電子技術総合研究所 (1978).
5) 黒川利明: LISP のデータ表現，情報処理，Vol. 17, No. 2 (1976).
6) 黒川利明: LISP 1.9 プログラミングシステム，情報処理，Vol. 17, No. 11 (1976).
7) 竹内郁雄: 第2回 Lisp コンテスト，情報処理，Vol. 20, No. 3 (1979).
8) MELCOM UTS/VS LISP 1.5 説明書，三菱電機株式会社計算機製作所ソフトウェア管理センタ (1975).
9) MELCOM INTERLISP 使用手引書，三菱電機株式会社計算機製作所ソフトウェア管理センタ (1980).
10) MELCOM UTS/VS バッチ処理説明書，三菱電機株式会社計算機製作所ソフトウェア管理センタ (1975).
11) Anderson, J. R.: Language Memory & Thought, Lawrence Erlbaum Associates, Inc., The Experimental psychology series (1976).
12) 拡張 LINGOL—自然言語処理のためのプログラミングシステム，電子技術総合研究所推論機構研究室 (1978).
13) 臿見達夫他: 拡張 LINGOL の n 進木への拡張，情報処理学会計算言語学研究会資料 22-1 (1980).
14) 臿見達夫: LISP 1.9 on MELCOM COSMO User's Manual, 東工大システム科学専攻市川研究室 (1981).


(昭和56年9月11日受付)