3X-6

8086をターゲットとしたREDUCE3.2の作成と評価

山本 強 青木 由直

北海道大学工学部

1. はじめに:

ここ数年来、AI関連のソフトウェアシステムの開発が活発化してきている。特に核言語としてのLISP,PROLOGについては多くの処理系が32bitマシンを搭載したワークステションをターゲットとしてインプリメントされている。また、IBM-PCに代表される超小型コンピュータをターゲットとした処理系も活発に行なわれているが、前者と比較するとどうしても実験用、あるいは教育用といった性格が強く、実用処理系とは言い難いものが多い。つまり、処理系はあるがアプリケーションレベルでは使用に耐えないということが言える。しかし、現時点の普及を考えるとPCクラスのハードウェアで動作する実用レベルの処理系が実現できればその影響力はより大きいと考えられる。

我々は以前、68000MPUをターゲットとして実用レベルの処理系を目指したLisp68K[1]を開発したが、その経験を活かして今回、8086を使用した一般的なパーソナルコンピュータをターゲットとした大規模プログラムを実行可能なコンパイラベースのLISP処理系,"AMI-Lisp"を試作した。また、アプリケーションの一例として大型機上の代表的数式処理システム,REDUCE3.2の全システムを移植する事ができたので報告する。

2. 基本方針:

AMI-Lispの開発の動機は現時点で入手可能ないくつかのLISPのアプリケーションソフトウェア(REDUCE3,LINGOL,OPS5 etc)をPC上で実用レベルで稼働させようというものである。これらのソフトウェアはいずれも大型機、32ビットミニコンをターゲットとして開発されたものであり、必然的に大容量のシステムを仮定している。我々は以前、REDUCE3については68000へのインプリメントの経験があるが、その経験から、データ空間は比較的小さくてもプログラム空間が十分とれればインプリメントが可能である見通しを得ていた。同様に、LINGOLについてもデータベースを2次記憶に置く事でパーソナルマシン上にインプリメント可能であるとの報告も見られる。そこで、8086のアーキテクチュアからくる限定されたセル空間は妥協するとして、プログラム空間については最大使用可能な空間を与える事とし、他の手続き型言語の処理系と同様なコンパイラの形式を採用した。

つまり、AMI-LISP本体はインタープリタで供給されるが、それ自体にはコンパイラは含まれず、別の実行モジュールとして提供される。自明な事であるが、コンパイラ自体はAMI-LISPで書かれており、実行モジュールはセルフコンパイルの結果でもある。通常のLispコンパイラは関数単位の1PASS型のコンパイルを行なう事が多い。これは開発環境としては望ましいが実行環境として見た場合、関数呼び出しが常にsymbolを経由するため速度及びコード効率が低下する欠点があった。また、コンパイラが常駐せねばならず、少ない空間が有効に使われない欠点もある。AMI-Lispの採用したコンパイラは独立したユーティリティとして提供される2PASS型のコンパイラであり、インタラクティブなコンパイルは出来ないが得られるコードは関数呼び出しについてはマシンコードレベルでリンクされており、実行速度およびコード量について改善されたものとなっている。その結果REDUCE3.2のような1MBを越える大規模プログラムでもPC上に移植することが可能となった。AMI-Lispでのコンパイルのフローを図1に示す。最終的なコンパイル結果はLISPの常駐関数ライブラリとリンクされMS-DOS上の.EXE形式の実行可能ファイルとして得られる。

3. 基本仕様:

AMI-LISPはREDUCEの移植が念頭にあったため、Utah大版のStandard Lispの仕様に準じている。基本データタイプはAtomicなデータとして整数(9ビット、19ビット、32ビット),実数(64ビット),文字列,1次元ベクタ,ID(最大4096個)を持つ。また、定義できる外部関数は最大1024個である。Cons CellはIDと足して30Kである。処理系の記述はMS-DOS上のMASM(マクロアセンブラ)とLattice Cで行なわれており、MS-DOS/PC-DOSが動作するPCであれば実行可能である。セグメントのレイアウ

Implementation and Evaluation of REDUCE3.2 for the 8086 CPU

Tsuyshi YAMAMOTO, Yoshinao AOKI

HOKKAIDO Univ.

トを図2に示すが、最低256KBの記憶空間を要求し、大規模なプログラムがコンパイルされた場合はさらにセグメントが追加される。

4. REDUCE3.2の評価:

IBMマシン用のREDUCE3.2をFM-16Bでコンパイルした結果とその実行結果について述べる。REDUCE3.2はRLISP形式で記述されたソースが1MB以上に及ぶため、AMI-Lisp単独ではゼネレートできず、RLISP-S Lispトランスレーションは UNIX版のREDUCE3.2を使用し、Lispレベルのソースから AMI-Lispによってゼネレートを行なった。ゼネレートされた実行形式のREDUCE3.2はMS-DOSのEXE形式で500KB程であり、データ空間を含めても640KBのMS-DOSで実行可能な規模であった。また、システム立ち上げ後のフリーセルは約22Kセルであり、イニシャライズによって消費されたセル数は約4Kセル程であった。これはコンパイラのオプションとしてトップレベルから見える必要のある関数と無いものを陽に指定し、不要なsymbolの発生を押えた結果である。その結果、小型処理系であるにも関らず表1に示すような計算容量と実行速度が確認された。

5. むすび:

小型システムを対象としたコンパイラベースのLisp処理系の開発について報告した。得られた処理系は処理速度の点では従来の中型機に匹敵することが確認された。しかし、CPUアーキテクチュアの問題から自由セル数が32K程度に限定されるため応用が限定される不安があったが、REDUCE3.2を移植して実行してみた結果、この程度の規模でも極めて現実的である事が確認できた。

文献

1. 山本他"68000用SLISP処理系の開発とREDUCE 3.1の移植",第31回情報処理学会全国大会8E-5, 1985-9月

2. 山本他"MS-DOS上のコンパイラベースのLisp処理系の開発とREDUCE3.2の移植"情報処理学会記号処理研究会報告,1986-6月

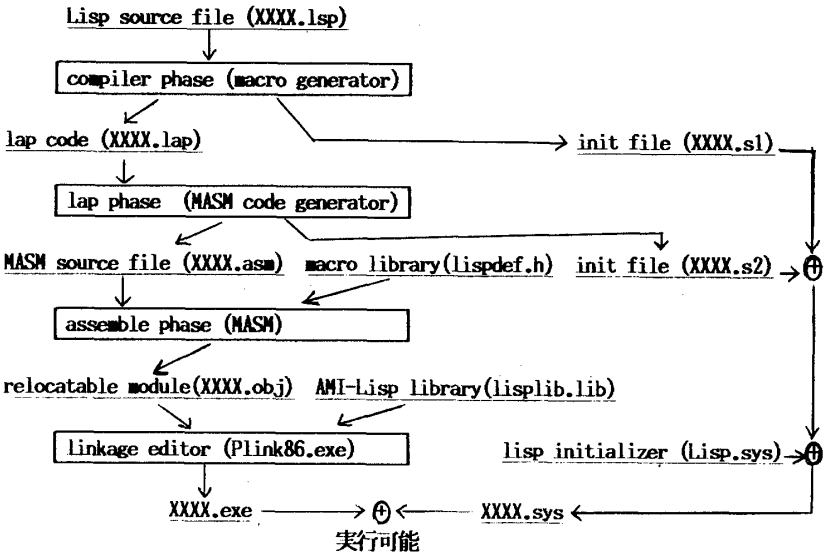

図1. AMI-Lispコンパイルフロー

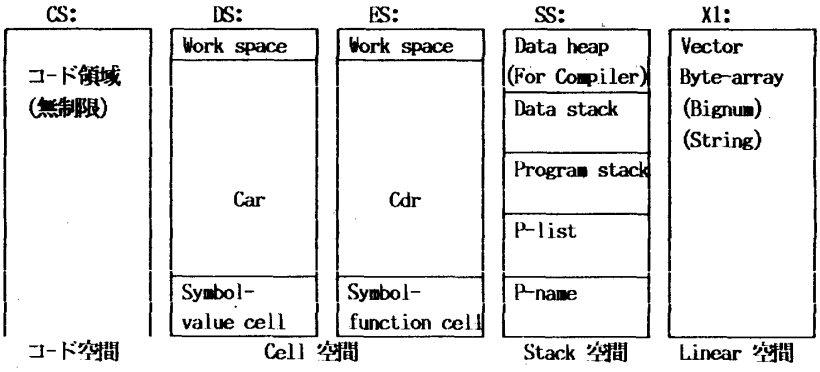

図2. AMI-Lispセグメント構造

表1. REDUCE3.2の容量限界と実行速度

A: 行列計算 (要素は未代入の変数)

行列次数	M*M$	M*M*M$	DET M$	M**(-1)$
5×5	2.1	12.0	2.4	379.0
6×6	3.3	22.2	6.3	*****
7×7	5.5	39.1	*****	*****
8×8	8.9	67.8	*****	*****
12×12	43.0	*****	*****	*****

B: 多項式の展開

演算	実行時間(秒)
(X1+X2+ . . . +X10)**4	7.5
(X1+X2+ . . . +X10)**5	23.0
(X1+X2+ . . . +X10)**6	*****(容量不足)
(X1+X2+ . . . +X5)**11	53.0
(X1+X2+ . . . +X5)**12	79.0
(X1+X2+ . . . +X5)**13	*****(容量不足)

注: FM16βにて計測