国際標準 LISP 言語 ISLISP のインタプリタおよびコンパイラ

6U－2

泉 信人　伊藤 貴康
東北大学 大学院 情報科学研究科

1 はじめに

Lisp 言語の ISO 標準として 1997 年 5 月に ISLISP が制定された [1]. ISLISP のインタプリタとコンパイラを作成し，これら処理系の評価を行ったので報告する．ISLISP のインタプリタとして OpenLisp[3] が存在するが，試作インタプリタは OpenLisp よりも高速であり，コンパイラはインタプリタより高速である．

2 ISLISP の概要

Lisp 言語の ISO 標準として制定された ISLISP は日本が提案した核言語を基に設計されたものである [2]. ISLISP は，SCHEME 並みにコンパクトな COMMON LISP 系の言語でオブジェクト指向機能を備えている．

ISLISP のプログラムは ISLISP テキストと呼ばれ，それは評価形式(リテラル，識別子，複合形式)からなる．複合形式は特殊形式，定義形式，関数適用形式，マクロ形式に分けられる．

ISLISP は 6 つの**名前空間**(変数，動的変数，関数，クラス，ブロック，タグ)をもつ多重名前空間方式を採用している．変数束縛は基本的に静的束縛であるが，(dynamic *name*)と明記することにより動的変数を扱うことができる．

ISLISP の**オブジェクト指向**は，COMMON LISP の CLOS を参考にしながらも非常にコンパクトに設計されている．クラスは，クラス名，スーパークラスのリスト，スロット指定，クラス任意機能からなる．ユーザが新しくクラスを定義する時には defclass 定義形式を用いる．包括関数およびメソッドは，それぞれ，defgeneric, defmethod によって定義し，オブジェクトの生成には包括関数 create を使用する．

3 インタプリタおよびコンパイラの概要

試作インタプリタは入力されたプログラムを直接解釈実行するのではなく，一度中間コードに変換してから解釈実行する．この中間コードを C 言語のプログラムに変換し，その後，C 言語のコンパイラでコンパイルしてより高速実行することが可能である．また，処理系全体は ANSI C で記述されており，移植性にも優れている．

中間コードは基本的に後置コードであり，試作した処理系では，リテラルオブジェクトをスタックに積む命令，ISLISP の定義済みの関数に対応する命令，ユーザ定義の関数を実行する命令など 260 種類以上の命令を使用している．

プログラムテキストの**関数オブジェクトへの変換**は次の 4 つのフェーズからなる．

1) テキストからリストへの変換
2) リスト構造から中間コードリストへの変換
3) 中間コードリストから関数オブジェクトの作成
4) 最適化

局所的な束縛が更に局所的な関数によって参照される場合にはスコープが閉じても束縛の値を残しておく必要があり，内部表現の環境オブジェクトに束縛の値を保持する．これ以外の場合はスコープの消失と共に値が参照されないのでスタック上で扱い，環境オブジェクトの作成コストや参照時の環境オブジェクトを辿るコストを削減している．

包括関数とは，引数のクラスによって呼び出し時の振る舞いが異なる関数である．包括関数が保持するメソッドの中から引数に対して適用可能なメソッドを選択する．適用可能なメソッドは引数のクラ

Interpreter and Compiler of the ISO Standard Lisp ISLISP
Nobuto Izumi, Takayasu Ito
Graduate School of Information Sciences, Tohoku University

ス優先度リストに従って並べられ，メソッド組み合わせ法に従って適用される．

クラス優先度リストはクラスとその上位クラスを優先度順に並べたものであり，クラス定義時に与えるスーパークラスのリストの順番で決定される．多重継承を許すISLISPでは，スーパークラスの種類が同じでもスーパークラス間の優先順位が異なることがある．**OpenLisp**では，クラス優先順位の決定法が正しく実現されていないため，メソッド呼び出しが正しく行われないことがある．試作処理系では，包括関数の呼び出し時に，包括関数オブジェクトが保持している全ての適用可能なメソッドを優先度順に並べた実行用の包括関数オブジェクトを作成し，メソッド呼び出しが正しく行われるよう実現している．なお，実行用包括関数オブジェクトを適用の度に作成するのは無駄であるので，一度作成したものは表に登録するなどの処理を行い包括関数の効率良い処理を実現している．

Lisp オブジェクトは固定長のセルと任意の長さを切り出せる不定長データを組み合わせて構成されている．これら二つの領域を使い分けGCによる実行の中断時間を短縮している．

コンパイラはインタプリタ内で実装されている関数 compile を呼び出し，コンパイル結果を C 言語のプログラムファイルとして出力する．コンパイルする関数オブジェクトの使用している全てのオブジェクトに対し環境初期化関数や中間コードに対応する関数を作成する．メモリ管理，リーダ，ライタなどはインタプリタと共通のソースを使用する．

4 実験評価

現在，試作処理系は PC(WindowsNT4.0) や WS(Solaris2.5.1)上などで動作している．表 1 に Gabriel Benchmark[4]，表 2 に包括関数を使用したプログラムの PC 上での実行時間(単位は秒)を示す．試作インタプリタが OpenLisp よりも高速なことや，コンパイラがインタプリタより高速であることが確認できる．なお，gqsort はオブジェクトの比較に包括関数を使用して quick-sort を行うが，OpenLisp は多重継承をしているクラスに対して正しいメソッドを呼び出せず，正しい結果が得られていない．

表 1: Gabriel Benchmarks

	OpenLisp (インタプリタ)	試作インタプリタ	試作コンパイラ
Tak	0.21	0.10	0.09
Stak	0.29	0.23	0.12
Ctak	0.39	0.22	0.15
Takl	1.53	1.11	0.28
Takr	0.26	0.16	0.16
Boyer	-	2.25	0.62
Browse	3.46	3.17	1.52
Destructive	0.62	0.51	0.30
Traverse-init	4.03	3.22	1.50
Traverse-run	21.67	15.47	6.97
Derivate	0.52	0.41	0.26
Dderivate	0.56	0.43	0.26
Div2-iter	0.42	0.29	0.16
Div2-rec	0.41	0.26	0.17
FFT	2.54	0.88	0.37
Puzzle	4.19	2.97	1.23
Triangle	51.69	42.22	12.97

表 2 :包括関数を使用したプログラム

	OpenLisp (インタプリタ)	試作インタプリタ	試作コンパイラ
gtak	0.32	0.26	0.20
gderiv	1.64	1.14	0.92
gqsort	-	0.80	0.60
LispInLisp (fib 15)	1.52	0.76	0.38

5 あとがき

試作した ISLISP インタプリタおよびコンパイラとそれらの評価実験結果について報告した．ISLISP インタプリタとしては OpenLisp が発表されているが，ISLISP のコンパイラとしては，筆者らの知る限り，本論文の試作処理系が初めての試みである．現在，型推論などを導入したより高性能なコンパイラを開発中である．

参考文献

1) ISO/IEC 13816:1997, Information technology - Programming languages, their environments and system software interfaces - Programming language ISLISP, (1997)

2) 伊藤 貴康: LISP 言語国際標準化と日本の貢献,「情報処理」38 巻 10 号, pp.932-937(1997)

3) Christian Jullien : OpenLisp, "http://www.ilog.fr:8001/Eligis/", (1998)

4) Richard P. Gabriel: Performance and Evaluation of Lisp Systems, MIT Press, (1985)