2E-5

EVLISマシンにおける並列Lispコンパイラの実現

安田弘幸，坂口寿和，三野雅仁，山田真一，斎藤年史，安井　裕
（大阪大学・工学部）

1．はじめに

我々は，リスト処理言語 Lisp の高速処理を目的とした Lisp の並列処理マシン ── EVLIS マシン[1)2)4)] ── の試作・研究を進めている．すでに，単体用のインタプリタ[3)]，コンパイラ[5)]および並列用インタプリタ[6)]が完成しており，Lisp コンテスト[7)]等の問題を並列および単体で実行しその高速性が示されている．現在，さらに高速な処理を実現するために，出力されるオブジェクトが並列で実行される Paralell Lisp コンパイラ(以下並列コンパイラという)のインプリメントを行っている．

本論文では，その並列コンパイラの並列化の着眼点を述べ，そのために追加した言語仕様および並列コンパイラの処理の概要，またその実行結果の一部を述べる．

2．並列化の着眼点

コンパイラでの並列化を論じるまえに，すでに報告した並列インタプリタでの処理アルゴリズムについて簡単に述べる．インタプリタ内での関数 evlis で分割された仕事をそれぞれプロセスと呼ぶ．これらのプロセスの処理を，複数台の EVAL-IIプロセッサが対等な立場で並列に実行する．つまり，EVLIS マシンの名前にあるように Lisp 関数の引数を並列に評価することになる．並列処理において副作用に伴う問題はインタプリタが自動的にコントロールし，解決する．すでに，並列化のためのオーバヘッドの減少を目的として並列処理の選択を可能とする関数タイプを導入し，ユーザが積極的に並列化を指示することによって処理効率を上げることに成功した[8)]．

さらに，本論文ではコンパイラのオブジェクトでの並列を関数の引数並列評価に適用することを実現した．コンパイラでは，プロセスの処理において並列処理のためのオーバヘッドの割合が，インタプリタのそれに比べてふえる可能性がある．そこで，今回試作したコンパイラでは，並列化をコントロールする指示をユーザによって指定できるような言語仕様にした．これによって，並列処理の効率をより向上させることが可能である．Lisp の副作用関数の処理のタイミングをもユーザがコントロールでき，コンパイラ側での調整処理が小さくなる．

3．言語仕様

基本的に Lisp 1.5 に準拠した EVLIS マシンの Lisp を使用する．2．で示した機能を実現するために，並列処理を期待する関数呼出しは次のような記述となる．

(※ <評価ｺﾝﾄﾛｰﾙﾘｽﾄ> <関数名> <引数>*)

<評価ｺﾝﾄﾛｰﾙﾘｽﾄ> ::= <順位評価ﾘｽﾄ>|<empty>
<順位評価ﾘｽﾄ> ::= (<順位ﾌﾟﾛｾｽ並び>)
<順位ﾌﾟﾛｾｽ並び> ::=<順位ﾌﾟﾛｾｽ>|
　　<順位ﾌﾟﾛｾｽ並び>･<順位ﾌﾟﾛｾｽ>
<順位ﾌﾟﾛｾｽ> ::= <引数番号>|<並列評価ﾘｽﾄ>
<並列評価ﾘｽﾄ> ::= (<並列ﾌﾟﾛｾｽ並び>)
<並列ﾌﾟﾛｾｽ並び> ::=<並列ﾌﾟﾛｾｽ>|
　　<並列ﾌﾟﾛｾｽ並び>･<並列ﾌﾟﾛｾｽ>
<並列ﾌﾟﾛｾｽ> ::= <引数番号>|<順位評価ﾘｽﾄ>
<引数番号> ::= 1|2|3|4|5|6|7|……

<引数番号> は与えられた引数に対して，左から右へ 1,2,3,… と順につけた番号である．ひとつの <評価ｺﾝﾄﾛｰﾙﾘｽﾄ> の中に現われる <引数番号> は重複してはいけない．また，存在しない <引数番号> を記述することはできない．

<順位ﾌﾟﾛｾｽ並び> は記述されている <順位ﾌﾟﾛｾｽ> に左から順に優先順位がついていることを意味する．副作用はこの順位に従って作用される．

<並列ﾌﾟﾛｾｽ並び> は記述されている <並列ﾌﾟﾛｾｽ> がすべて並列に評価され，そのプロセス間の優先順位はない．

<評価ｺﾝﾄﾛｰﾙﾘｽﾄ> が <empty> のときはすべての引数に対して左から右へ優先順位がついているとみなす．

{※ <評価ｺﾝﾄﾛｰﾙﾘｽﾄ>} 全体が記述されていないときは，引数は左から右へ順に逐次評価され，並列化されない．

[例] (※ (((1 2 4) 3)) fn arg1 arg2 arg3 arg4)

この例では，arg1,arg2,arg4 の順に引数が優先順位をもって評価され，その処理と並列に優先順位に関係なく arg3 が評価されることを示す．すべての値がそろった後，関数 fn が call される．arg1,arg2,arg4 の引数間の副作用はこの評価順に作用することが保証されている．

このように，関数に与えられる引数の評価順序をユーザーが任意に決めることが可能である．

Paralell Lisp Compiler for EVLIS Machine
Hiroyuki YASUDA, Toshikazu SAKAGUCHI, Masahito MINO, Shinichi YAMADA,
Toshifumi SAITO, Hiroshi YASUI
OSAKA, Univ.

4. EVLIS マシンおよび並列コンパイラの概要

4.1 EVLIS マシンの概要

EVLIS マシンのハードウェア構成について簡単に説明する.

・EVAL-IIプロセッサ[2)4)]

マイクロプログラム制御方式の高速リスト処理プロセッサで，現在2台実装している．ハードウェアスタック等に用いるスクラッチパッドメモリ(SM)を備えている．制御記憶(WCS)は 8K words あり，I/Oプロセッサを通して書き換え可能である．

・メインメモリ(MM)

リストデータ，フレーム等の共有データを格納し，プロセッサ間通信にも使用している．現在 8 banks 実装し，総容量は 32K wordsである．

・q-buffer

共有の高速 FIFO メモリで，処理待ちフレームの queue を保持している．

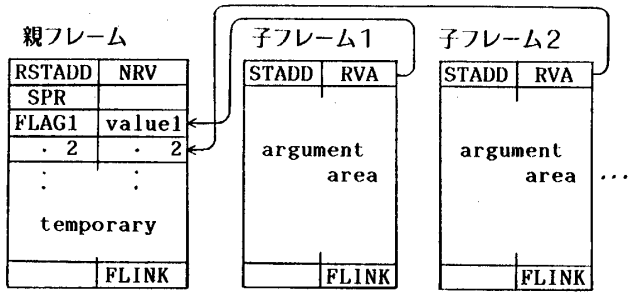
・I/O プロセッサ

LISP 入出力，EVAL-IIプロセッサのコントロール，ハードウェアのデバッグ支援等の機能をもつ．

4.2 並列コンパイラの概要

本コンパイラは，Lisp で記述されている．すでに稼働している単体用コンパイラをベースに，さらに並列処理のために以下の機能を加えている．

コンパイラは関数の引数並列評価を指示した〈評価ｺﾝﾄﾛｰﾙﾘｽﾄ〉のうちのひとつの〈並列評価ﾘｽﾄ〉に対して，ひとつの親フレームと，そのリストの中に現われる〈並列ﾌﾟﾛｾｽ〉の数に対応する子フレームとを MM に作成するコードをつくりだす(Fig.1)．作成された子フレームはq-bufferに登録され，処理待ちプロセッサがそのフレームを獲得して処理を開始する．親フレームを生成したプロセッサは，処理再開に必要な情報を親フレームに退避して，他の処理待ちフレームを獲得し処理する．処理の終った子フレームは，その値を親フレームに返し，親フレームの処理再開が可能なら，必要な情報をフレームよりもどして，処理を再開する．処理再開が不可能ならば新しいフレームの処理に移る．



RSTADD:restart address STADD:start address
NRV :number of value RVA :return value address
SPR :store of SPR FLAG :various flag
FLINK :extension frame pointer

Fig.1 フレームの構造

今回試作したコンパイラは，3．での言語仕様をほぼみたすように設計されている．副作用の自動管理に関しては，現在インプリメント中である．

5. 実行結果

この並列コンパイラでコンパイルし，その実行結果の一例として Lisp コンテスト[7)]等の問題 (Tarai関数，List-Tarai関数，Srev関数) およびフィボナッチ関数について，単体コンパイラでの実行時間とともに Table 1. に示す．ただし，共にマイクロコードレベルでの最適化は行っていない．この並列コンパイラでの実行時間はひとつの並列化の例であり，3．で述べた並列化の指示の与え方によってオブジェクトの動きが変化し，実行時間も減少する．

Table 1. 単体および並列コンパイラにおける実行時間

	単体ｺﾝﾊﾟｲﾗ(S)	並列ｺﾝﾊﾟｲﾗ(P)	比率(S/P)
Tarai-5	1630.3 ms	1026.0 ms	1.6
List-Tarai-4	188.1	110.3	1.7
Fibb(20)	126.9	64.2	1.98
Srev-6	14.3	10.7	1.3

測定時のEVAL-IIおよびMMのクロックはそれぞれ 100ns,61ns である．

6. まとめ

現在，副作用が起こったときのプロセスの自動管理，オブジェクト効率を上げるための最適化などを行っている．また，ユーザが特に意識しなくても自動的に高効率な並列処理が行える〈評価ｺﾝﾄﾛｰﾙﾘｽﾄ〉を出力するようなプリプロセッサの開発など，今後の課題のひとつである．

［参考文献］

1) 安井 裕 他：LISPでの並列処理における動特性とEVLISﾏｼﾝの構成，情処，記号処理資料10-4(1979)

2) 前川博俊 他：高速LISPﾏｼﾝとﾘｽﾄ処理ﾌﾟﾛｾｯｻEVAL II，情処論文誌，Vol.24,No.5,pp683-688(1983)

3) 斎藤年史 他：高速LISPﾏｼﾝとﾘｽﾄ処理ﾌﾟﾛｾｯｻEVAL II，情処論文誌，Vol.24,No.5,pp689-695(1983)

4) Maegawa,H. et al.：Fast LISP Machine and List-Evaluation Processor EVAL-II,JIP,Vol.8,No.2, pp121-126(1985)

5) 高橋俊樹 他：EVLISﾏｼﾝにおけるLISPｺﾝﾊﾟｲﾗ，情処第32回全大，4G-11(1986)

6) 西川 岳 他：EVLISﾏｼﾝの並列処理ｱﾙｺﾞﾘｽﾞﾑとそのｲﾝﾌﾟﾘﾒﾝﾃｰｼｮﾝ，情処第23回全大，4H-7(1981)

7) 奥野 博：第3回Lispｺﾝﾃｽﾄおよび第1回Prologｺﾝﾃｽﾄ報告，情処，記号処理資料13-4(1985)

8) 西開地秀和 他：EVLISﾏｼﾝのLISP並列処理における動特性，情処第29回全大，6B-1(1984)