4B-9

Ｐｒｏｌｏｇの動作特性に関する考察

新井　進　岸本　光弘　湯原　雅信　服部　彰
(富　士　通　研　究　所)

1．はじめに

本稿では、Lisp,Prolog 専用機であるＦＡＣＯＭα上のProlog コンパイラであるα-Prolog のコンパイルドコード実行時のアクセス・パターンをトレースによって測定し、その結果をもとに Prolog 処理系に向いたキャッシュメモリの構成について述べる。

尚、本処理系は第５世代コンピュータプロジェクトの一環としてＩＣＯＴの委託で開発したものである。

2．α－Ｐｒｏｌｏｇコンパイラの特徴

ＦＡＣＯＭαは、D.H.D.Warrenの抽象マシン命令セットと同等のレベルのマシン命令セットを持っている。メモリ・アクセスの面から見た両者の主な相違点は、頻繁にアクセスされる局所変数とローカル・スタックを、高速アクセスが可能なハードウェア・スタック上に割り当てていることである。そのために、コンパイルドコード実行時のアクセスは、表１に示すようにローカル・スタック（注：これ以降では、本来の意味でのローカル・スタックへのアクセスと、局所変数へのアクセスを一緒にしてローカル・スタックへのアクセスと呼ぶ。）に集中している。

領域	ローカルスタック	グローバルスタック	トレイルスタック	その他
比率	９４．９	３．５	０．４	１．３

表１　各データ領域へのアクセス回数比　(%)

3．アクセス・パターンの測定

Prologのコンパイルドコード実行時におけるアクセス・パターンを知るために、α-Prolog のコンパイルドコードが、ハードウェア・スタックと主記憶をどのようなパターンでアクセスするのかを測定した。

今回行った測定では、１回の実行での総アクセス回数が10^7 回にも達するため、各アクセスの履歴を直接記録することはせず、測定対象のプログラムを実行しながらキャッシュメモリのシュミレーションを並行して行った。

また、ＦＡＣＯＭαでは、主記憶とハードウェア・スタックとは、別の空間に配置されているが、今回は両者を同一の空間上に配置し、汎用マシンに近い条件でシュミレーションを行なった。

4．測定結果

キャッシュメモリの容量、ブロック長、連想レベルをパラメタとして、次の条件の下でシュミレーションを行った。

・リプレースメント方式　ＬＲＵ
・書き込み方式　コーピーバック方式
・キャッシングの対象　すべてのデータ領域

4.1　キャッシュメモリの容量とヒット率

図１にキャッシュメモリの容量と、ヒット率の関係を示す。（注：図中のミスヒット率の値は、すべて全アクセス回数に対するものである。）この図から、比較的小さなキャッシュでも十分なヒット率が得られることがわかる。これは、アクセス回数の大半を占めるローカル・スタック上のデータの局所性が高いためである。また、スタック・フレームの大きさ（α-Prolog では11語）の数倍の容量でヒット率が飽和していることから、一定の時間内に参照されるのは、高々数個のフレームに限られている、つまり時間的な局所性が高いことを示している。これは、Prologコンパイラが終端再帰呼び出しの最適化を行なっているために、参照するスタック・フレームの位置の変化が少ないためである。

それに対して、グローバル・スタックに対するアクセスは、絶対的な回数が少ないにもかかわらず、ローカル・スタックとほぼ同数のミスヒットを起している。これは、構造体、リスト等のデータがグーロバル・スタック上に散在していることと、他のデータ領域（ローカル・スタック、トレイル・スタック）とのキャッシュの衝突が生じるため、つまり空間的にも時間的にも局所性が乏しいためである。

4.2　ブロック長とヒット率

図２にブロック長とヒット率の関係を示す。一般的にはブロック長を大きくすることでデータの先読み効果が生じ

A Study on Prolog's Dynamic Characteristics

Susumu Arai, Mitsuhiro Kishimoto, Masanobu Yuhara, Akira Hattori
Fujitsu Laboratories Ltd.

キャッシュのヒット率は向上するが、先に述べたようにα-Prolog コンパイラでは、データの空間的な局所性は高々スタック・フレームの大きさ程度しかないため、フレームの大きさを越えたブロック長の増加は必ずしもヒット率の向上には結び付かない。むしろ、小容量のキャッシュにおいては、エントリ数の減少によってキャッシュの衝突の可能性が増大し、ヒット率の低下を招く場合も生じる。

4.3 連想レベルとヒット率

図3に連想レベルと、ヒット率の関係を示す。ローカル・スタックのアクセスは、すでに述べたように時間的な局所性が大きいために、連想レベルを増しても、キャッシュのヒット率の増加はごく僅かである。

それに対して、時間的な局所性の乏しいクローバル・スタック上のデータのアクセスでは、他のデータ領域（主にローカル・スタック）との衝突が避けられるようになるため、ヒット率はかなり向上する。

5. まとめ

以上述べたように、Prologのコンパイルドコードの実行時のメモリ・アクセスは、極めて高い時間的局所性を有するので、連想レベル、ブロック長の双方が小さい単純な構成のキャッシュ・メモリでも、十分なヒット率を得ることができることがわかった。

また、アクセス頻度の高いローカル・スタックだけをキャッシングするのであれば、大きなブロック長をもつキャッシュメモリでもキャッシュの衝突は、特に問題にならないため、ＦＡＣＯＭαのハードウェア・スタック（キャッシュ容量＝８Ｋ語、ブロック長＝２Ｋ語、フルアソシエイティブのキャッシュメモリ）が、少ないハードウェア量で十分な性能を得るのに有効な手段であることを確認できた。

今回のシュミレーションでは、小規模なプログラムを測定の対象としたが、より正確なデータを得るためには、さらに大規模なプログラムでのシュミレーションを行なう必要がある。

謝辞　日頃御指導頂く棚橋部長、並びに研究室諸兄、そしてデータ収集に協力して下さった細井聡君に感謝します。

参考文献

[1] D.H.D.Warren:AN ABSTRACT PROLOG INSTRUCTION SET. Tech Note 309, AIC SRI International.

[2] 岸本他: Prologコンパイラの設計と評価, The Logic Programing Conference '85

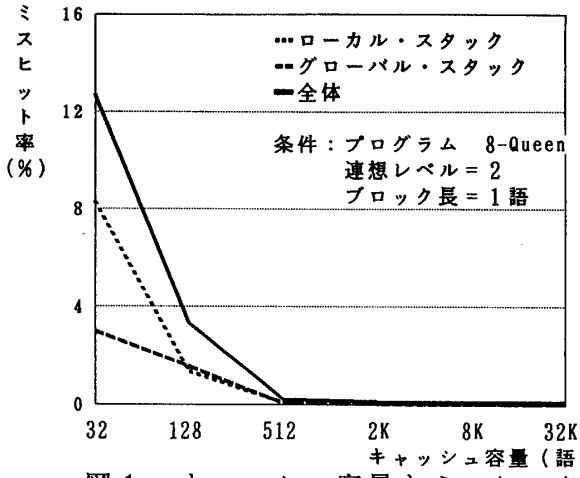

図1　キャッシュ容量とミスヒット率

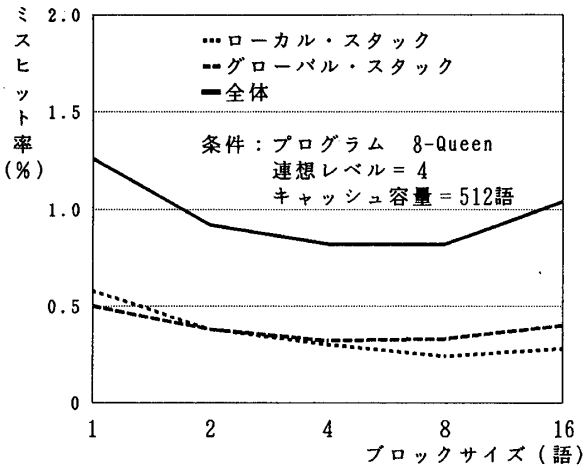

図2　ブロック長とミスヒット率

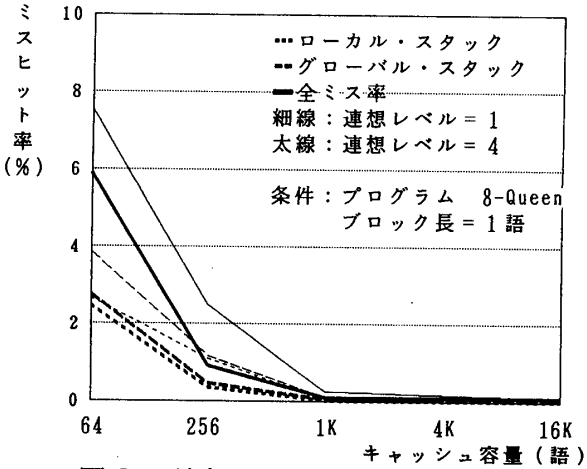

図3　連想レベルとミスヒット率