

UNIXワークステーション上での最適化PROLOGインタプリタの性能評価

7Q-3

碓崎 賢一

九州工業大学 情報工学部

1 はじめに

PROLOG が広く利用されるようになるにつれて、その処理速度とメモリ効率の向上が強く望まれるようになってきている。PROLOG は、コンパイルされたプログラムにおいてもインタプリタが多用される傾向があり、PROLOG 処理系の性能を向上させるためには、コンパイラだけでなくインタプリタの性能の向上が必要不可欠である。このような観点から開発されたのが最適化 PROLOG インタプリタ^[1]である。

最適化 PROLOG インタプリタは、インタプリタでありながら WAM^[2]と基本的に同じ実現方式を採用した上で、さらにいくつかの新たな最適化手法を導入し、処理速度とメモリ効率を向上させた処理系である。最適化 PROLOG インタプリタの開発と評価は、今までパーソナルコンピュータ上で行ってきたが、このほど UNIX ワークステーション上に移植したので、ワークステーション上で行った性能評価の結果について報告する。

2 最適化手法の概要と性能評価

最適化 PROLOG インタプリタは C 言語で記述されており、MS-DOS を使用した 16 ビットパーソナルコンピュータと、UNIX を使用した 32 ビットワークステーションで稼働している。移植性の高い同一のソースプログラムがパーソナルコンピュータとワークステーションで使用されているために、両者の処理系が同一のシンタックスとセマンティックスを持つという大きな特徴がある。

最適化 PROLOG インタプリタには、各種の最適化手法が取り入れられており、処理速度とメモリ効率の向上が図られている。最適化手法の詳細は文献 1, 3 に示されているので、本報告では、最適化 PROLOG インタプリタで採用している手法の一部分についての概要とその評価を示す。

2.1 ベンチマークテストの処理速度

代表的な 3 種類のベンチマークプログラムを使用して処理速度の測定を行った。nrev (nreverse) と qsort は、連続して 100 回実行させ、その実行時間を平均して処理時間を得ている。queen8 は、すべての解を求めさせて処理時間

を得ている。また、他の処理系との比較のために Quintus Prolog のインタプリタの処理速度を測定した。

表 1 に示されるように、最適化 PROLOG インタプリタは、Sun4 上で最高 30K LIPS と非常に高い処理速度を達成している。この処理速度は、ICOT の PROLOG マシン PSI 1 に匹敵するもので、本インタプリタに採用されている各種の最適化手法の有効性が示されている。また、Sun4 上では、Quintus Prolog の 4 倍程度の処理速度が実現されているのが明らかになっている。さらに、インタプリタのみの処理速度で比較した場合には、3 MIPS のマシン (LUNA) 上の本インタプリタが、10 MIPS のマシン (Sun4) 上の従来のインタプリタと同等の性能を持つことが示されている。

2.2 述語の動特性を考慮したインデッキング手法

最適化 PROLOG インタプリタでは、インデッキング用の情報をプログラムの読み込み時に作成する方式と、述語の動特性の宣言を利用して選択点の生成を抑制する方式を採用することによって、インタプリタでありながら効率のよいインデッキングを実現している。動特性には、動的と静的があり、実行時に変更される可能性がある述語は前者に、変更されない述語は後者に分類され宣言が付加される。静的と宣言された述語は実行時の節の追加に備えて選択点を残しておく必要がないため、不必要な選択点を除去する最適化を行なうことができる。

従来の処理系のなかには、動特性の宣言を行わずに選択点の生成を抑制しているものもあるが、そのような処理系では、代替節の利用に関するセマンティックスに一貫性がない^[1]という問題があった。

表 1 に示した処理速度は静的述語として実行して得られたものであるが、最適化の効果を明らかにするために、動的述語として宣言を行い、選択点に関する最適化を行わない場合の処理速度を測定したので表 2 に示す。表 2 に示している比率は、動的述語としての処理速度に対する静的述語としての処理速度 (表 1) の向上比である。

表 2 に示されるように動的述語として実行した場合に比べ、静的述語として実行した場合には、3 種類のすべての

表 1 ベンチマークテストの処理速度

述語	最適化 PROLOG インタプリタ						Quintus Prolog (インタプリタ)			
	LUNA SX9100/DT37		Sun3/260		Sun4/260		Sun3/260		Sun4/260	
	m sec	LIPS	m sec	LIPS	m sec	LIPS	m sec	LIPS	m sec	LIPS
nrev	68.7	7.2K	45.5	10.8K	16.3	30.4K	113.5	4.4K	68.2	7.3K
qsort	119.5	5.1K	81.5	7.5K	31.2	19.5K	174.2	3.5K	108.2	5.6K
queen8	15,500	----	10,900	----	4,200	----	24,000	----	14,700	----

Performance Evaluation of the Optimized PROLOG Interpreter on UNIX Workstations

Ken'ichi KAKIZAKI

Kyushu Institute of Technology

表2 動的述語としての処理速度

述語	LUNA		Sun3/260		Sun4/260	
	m sec	比率	m sec	比率	m sec	比率
nrev	83.2	1.21	59.2	1.30	23.8	1.46
qsort	123.3	1.03	86.0	1.06	33.3	1.07
queen8	16,000	1.03	11,300	1.04	4,300	1.02

プログラムで処理速度が向上しており、選択点に関する最適化の効果が示されている。また、Sun4 における nrev や qsort の速度向上率が大きいことが示されている。これは、RISC マシンではメモリの参照頻度の低下による処理速度の向上率が大きいこと、最適化によるメモリ効率の向上が処理速度の向上にも大きく影響したものと考えられる。

動特性の違いによるメモリの使用量を表3に示す。表3には、動特性の違いによって影響を受ける制御スタックとトレイルスタックに関して示した。

表3 スタック領域のメモリ使用量
(単位は bytes)

述語	静的述語		動的述語	
	制御	トレイル	制御	トレイル
nrev	0.7K	0	20.4K	1.8K
qsort	0.0K	0	13.5K	1.2K
queen8	0.9K	0.1K	4.3K	0.2K

表3では、静的述語として実行した場合の制御スタックのメモリ使用量は、最高の qsort で 270 分の 1 以下、最低の queen8 で 5 分の 1 になることが示されており、メモリ効率が大幅に向上していることが明らかになっている。

データベースの操作性能の評価を表4に示す。この評価は、節の登録時や除去時に行っているインデッキング用情報の操作が処理速度へ与える影響を調べるために行ったものである。使用したプログラムは、整数値を引数に持つ 1 引数の複合項をファクトとしてデータベースに登録、除去するもので、処理は、再帰的なループを利用して 100 個のファクトをまず登録し、次に retract を使用して 1 つずつ削除している。処理速度は、asserta を使用してデータベースの最初に登録する場合と、assertz を使用してデータベースの最後に登録する場合を測定している。

表4 データベースの操作性能
(単位は m sec)

節の追加位置	最適化 PROLOG インタプリタ			Quintus Prolog			
				インタプリタ		コンパイラ	
	LUNA	Sun3	Sun4	Sun3	Sun4	Sun3	Sun4
最初	74	57	22	355	255	205	152
最後	104	83	39	353	255	200	152

表4に示されるように、データベースの操作に関しても、最適化 PROLOG インタプリタは従来のインタプリタと比較して 4 ～ 12 倍高速であり、インデッキング用の情報の

作成による性能低下はみられない。さらに、コンパイルされたプログラムと比較しても、3 ～ 7 倍高速であり、データベースを多用する処理を高速化するためには、インタプリタの処理速度の向上が不可欠であることが示されている。

最適化 PROLOG インタプリタで、データベースの最後に節を追加する操作が遅いのは、データベースの最後の節を操作する場合であっても、最初の節から次の節を指すポインタを順にたどっていく方式を採用しているためである。この方式では、データベースの最後に登録する節が増加すると処理速度が低下していくと考えられるが、節の数が十分に多い 100 個の場合でも処理時間は 2 倍以下におさまっているため、実用上十分な処理速度であると考えられる。

2.3 変数の単一化処理の最適化

最適化 PROLOG インタプリタは、変数の出現状況を解析し、不必要な変数の単一化処理を省略する最適化手法^[2]を採用している。この最適化手法の効果を明らかにするために、最適化を行わなかった場合の処理速度を表5に示す。

表5 変数の単一化の最適化を行わない場合の処理速度

述語	LUNA		Sun3/260		Sun4/260	
	LIPS	比率	LIPS	比率	LIPS	比率
nrev	5.5K	1.31	8.7K	1.24	25.2K	1.19
qsort	4.6K	1.11	6.9K	1.09	18.7K	1.04

表5に示している比率は、最適化を行わなかった場合の処理速度に対する最適化を行った場合(表1)の処理速度の向上比である。レジスタ割当の最適化を行い、単一化処理を省略することによって最高で 1.31 倍ほど処理速度が向上していることが示されている。Sun4 の処理速度の向上が他の 2 つのマシンに比べて小さい理由は、次のように考えられる。nrev や qsort で省略される単一化処理は、基本的に move 命令に置き換えられる単純な処理である。最適化 PROLOG インタプリタでは、この処理を実行するためにサブルーチンと呼んでいるが、一般的にはサブルーチンコールには比較的大きな処理時間がかかる。したがって、最適化を行ったときの処理速度の向上の要因の中には、単一化処理自体を省略することによって得られる性能の向上とともに、サブルーチンコールを除去することによる性能の向上が含まれている。しかしながら、Sun4 はレジスタバンクの切り替えを利用した高速なサブルーチンコール機構を持っているために、サブルーチンコールを除去することによる処理速度の向上が少なかったものと考えられる。

3 まとめ

最適化 PROLOG インタプリタの評価を行い、10 MIPS の性能を持つワークステーション上では、30K LIPS 以上の処理速度を実現できることを確認した。また、従来のインタプリタと比較して処理速度が 4 倍程度になり、メモリ効率も大幅に改善されることを示した。

参考文献

- [1] 碓崎 他：最適化 PROLOG インタプリタの構成法，情報処理学会，記号処理研究会資料，48-4，(1988)。
- [2] Warren, D.H.D.: An Abstract Prolog Instruction Set, SRI International, Tech. Note 309, (1983)。
- [3] 碓崎 他：最適化 PROLOG インタプリタの性能評価，情報処理学会，第 38 回全国大会講演論文集，pp.1021-1022, (1989)。