1L－9

並列粒度調整機能を組み込んだ SISALコンパイラの設計と実装

高畠 志泰† 大澤 範高† 弓場 敏嗣† 佐藤 三久‡ 山口 喜教§
† 電気通信大学大学院 情報システム学研究科 ‡ 新情報処理開発機構 § 電子技術総合研究所

1 はじめに

分散記憶型並列計算機において、使用する要素プロセッサ数を多くして、各要素プロセッサでの処理時間を減らしても、プログラムの実行時間が必ずしも速くなるとは限らない。これは、要素プロセッサ数が増えると、一般的には、要素プロセッサ間での通信量が増加するからである。

従来の並列化コンパイラには、各要素プロセッサへの処理をスケジューリングするのに、通信時間と通信量を共に考慮することはされていなかった。本稿では、与えられた並列プログラムにおいて、静的に解析可能な並列性を考慮し、プログラム全体の処理時間を小さくする最適な要素プロセッサ数と、そのときの処理粒度の大きさを求める並列粒度調整方法を提案し、コンパイラに組み込む。

コンパイラの設計と実装にあたり、言語は、関数型言語SISAL(Streams and Iteration in a Single Assignment Language)[2]を使う。対象とする並列計算機は、高並列データ駆動型計算機EM-Xを使用する。

2 関数型言語 SISAL

SISALは、単一代入規則をもち、ループの実現方法や、ストリームという構造体も扱えるという特徴をもつ汎用性のある並列処理記述に適した言語である。この言語の単一代入性、関数的性質により、各要素プロセッサにおける処理粒度(スレッド)の大きさが調整可能となる。

3 並列粒度調整方法

本研究では、SISALプログラムをEM-X用オブジェクトプログラムにコンパイルするときに、各要素プロセッサで並列実行される並列粒度の調整を行う。ここで粒度とは、要素プロセッサ間の通信を行わずに並列で実行することのできる処理の単位(スレッド)のことであり、粒度調整とは、そのスレッドの処理時間(粒度の大きさ)を調整することを意味する。

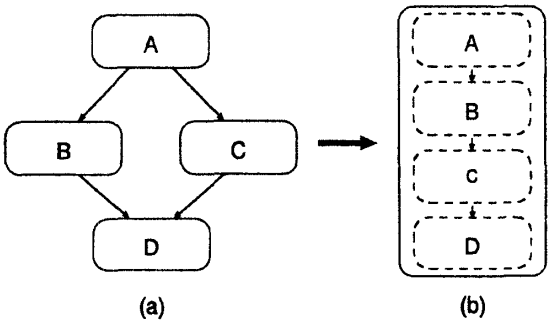

図1: 並列粒度調整

3.1 logP モデル

通信時間を考慮するために、並列計算機の評価モデルであるLogPモデル[1]を使用する。LogPモデルは、並列計算機上での並列プログラムの実行時間評価するものである。並列計算機の要素プロセッサ間の通信時間を以下の4つのパラメータを使って表す。

L：通信遅延。

o：送受信オーバヘッド。

g：送受信できる最小時間間隔(バンド幅)。

P：要素プロセッサの数。

ここで、EM-Xに実装するため、LogPモデルを拡張し、スレッドの生成場所を決める時間と生成時間も考慮に入れる。

3.2 粒度調整の方法

図1の(a)のような並列に実行可能な部分について考える。AからB,CへとB,CからDへの通信時間は、logPモデルで計算できる。A,B,C,Dの実行時間が、コンパイル時に静的にきめることができると、このプログラムの実行時間がわかる。このとき、BとCを並列に実行する方が良いか、または、逐次に実行する方が良いかは、コンパイル時に知ることができる。この考え方を基本にし、A,B,C,Dの実行方法を決定し逐次の

Design the SISAL Compiler with Parallel Granularity Tuning Function

Motoyasu Takabatake† Noritaka Osawa† Toshitsugu Yuba† Mitsuhisa Sato‡ Yoshinori Yamaguchi§
†Graduate School of Information Systems, The University of Electro-Communications
‡Real World Computing Partnership
§Electro-Technical Laboratory

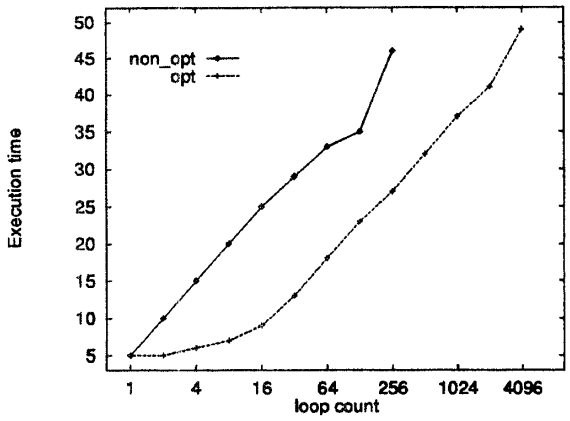

図 2: ループの粒度調整効果

表 1: ループの粒度調整効果

ループ回数 (N)	8	16	32	64	128
粒度調整前 (μsec)	20	25	29	33	35
粒度調整後 (μsec)	7	9	13	18	23
比率 (%)	35	36	45	55	66

場合は1つにまとめ、再帰的に用いて並列粒度を調整する。[3]

本稿で提案する粒度調整を行う部分は、1) 並列ループと、2) 関数呼出しである。並列ループの場合は、いくつかのイタレーションを1つのスレッドとして生成するようにし、1つのスレッドに入れるイタレーションの数で粒度を調整する。関数呼出しの場合は、同時に2つ以上の関数呼出しをすることができるもので、1つの関数を1つのスレッドとして、スレッドを生成するときに、すべてのスレッドを自分の要素プロセッサへか、または、別の要素プロセッサへ割り当てることで粒度を調整する。

4 予備的評価

図2、表1は、1から N までの和を求めるプログラムにおける並列ループの並列粒度調整の効果を示したものである。粒度調整を行わずにループをできるだけ多くのの要素プロセッサに割り当てたときとの比較である。今回の実験では、ループボディでの処理が小さいために、粒度調整の効果も大きかった。ループ回数が小さいときに効果が大きいのは、使用する要素プロセッサの数が少ないために、通信量が少ないからである。ループ回数が増加すると使用する要素プロセッサの数が増えるために、粒度調整の効果は小さくなるが、効果がなくなることはなく、約65% で実行できる。

5 おわりに

並列ループの粒度調整について、本稿で提案する粒度調整方法の有効性を示した。関数呼出しについても同様に有効性が得られることが分かっている。また、並列ループの粒度調整方法の場合は、他の分散記憶型並列計算機にも応用できる。

今後の課題として、SISAL コンパイラの実装や、実用のプログラムによる並列粒度調整の効果とその評価、並列粒度調整の適用範囲の拡大が挙げられる。

SISAL プログラムでは、スレッドの生成場所や、データの分散方法を記述しないので、全てコンパイラに任される。そのため、最適なスレッドの生成場所の決定方法や、データの分散方法をプログラムからどのように決めるかという問題がある。

実用プログラムにおける並列粒度調整の評価においては、プログラムには、並列動作する部分が、コンパイル時に実行時間が特定できないときがある。このようなプログラムでは、本稿の粒度調整は使えない。しかし、静的に特定できる部分については効果は期待できる。また、他の一般的な最適化とトレードオフにならないので、あらゆる場合に適応できる。

並列粒度調整の適用範囲は、現在、並列ループと関数呼出しの2つだけである。ソフトウェアパイプラインなどの他の並列動作可能な部分についても粒度調整を行い、コンパイル時に最適な処理粒度を決定する方法を検討している。並列ループと関数呼出しは、プログラムから明示的に分かるが、それ以外の場合は、プログラムから並列性を抽出する方法も考慮しなければならない。

参考文献

[1] D. Culler, R. Karp, D. Patterson, A. Sahay, K. Schauser., E. Santos, R. Subramonian, and T. V. Eicken. LogP: Towords a realistic model of parallel computation. *Proc. the 4th ACM SIGPLAN Symposium on Principles & Practice of Parallel Programming*, pp. 1–12, 1993.

[2] J. McGraw, S. Skedzielewski, S. Allan, R. Oldehoeft, J. Glauert, C. Kirkham, B. Noyce, and R. Thomas. SISAL: Streams and iteration in a single assignment language: Language reference manual: Language version 1.2. *Manual M-146, Rev. 1, Lawrence Livermore National Laboratory and Colorado State University*, 1985.

[3] 高畠, 大澤, 弓場. SISAL コンパイラへの並列粒度調整機能の組み込み. 情報処理学会研究報告,96-HPC-62, Vol. 96, No. 81, pp. 75–80, 1996.