1P−6

並列ループ解析ツールの実現

福田 宗弘
日本アイ・ビー・エム株式会社　東京基礎研究所

１．はじめに

共有メモリ型マルチプロセッサの実用化にともない、この上でDOループを並列に実行することが試みられている。より多くのDOループに対して並列処理の効果を得るために、ループの並列化技術の改善に加えて、並列支援機構を検討する必要がある。そこで、並列処理ワークステーション TOP-1[1]の上で、行列演算、ベンチマーク等のパッケージを並列実行し、その結果を用いて並列DOループのオーバヘッドを解析し、並列支援方法、機構等を検討する。現在、TOP-1上でDOループの並列実行と実行時のオーバヘッドの解析を行うツールCFORを開発している。本稿では、この解析ツールCFORの実現方法について述べるとともに、CFORを用いたリバモア・ループ[2]の並列実行例を示す。

２．解析方針

２．１ 解析環境

キャッシュ・メモリを装備した10台のプロセッサからなる共有メモリ型マルチプロセッサ TOP-1の上で、複数のプロセスによりDOループを並列実行し、オーバヘッドの解析を試みる。並列ループのDO変数の制御、繰返し間の同期制御に関して種々の方法が提案されている[3]。各方法により、実行時における並列ループの動作が異なる。そこで、並列ループの制御方法を組み合わせてその動作を変化させ、それぞれに関して実行時のオーバヘッドを測定する。

２．２ ループ制御

DO変数、繰返し間同期、DOスパンの各制御方法を組み合わせることにより、DOループの並列実行に変化を与える。

(1) DO変数の制御

各プロセッサが実行すべき繰返しを決定する方法として、プリスケジューリング、および、セルフスケジューリングを使用する。前者は、並列実行に先立って、予め各プロセッサに実行すべき繰返しを割り当てる。後者は、実行時に各プロセッサがFetch and AddによってDO変数を取り合い、実行すべき繰返しを決定する。

(2) 繰返し間の同期制御

繰返し間のデータの依存関係を保持する同期命令として、Send-Wait命令とTestset-Test命令を用意する。前者は、Send命令が他の繰返しの実行を再開させ、Wait命令が他の繰返しによるsend命令発行まで実行を一時停止する。後者は、Midkiffらによって提案された命令である[4]。Testset(r)命令は、共有メモリrの値が１つ前の繰返し番号になるまで実行を停止し、条件が満足されるとrの値を自分の繰返し番号に書き換え、実行を再開する。Test(r, d)命令は、rの値がdだけ前の繰返し番号になるまで、実行を待ち合わせる。

(3) DOスパンの制御

プログラムに記述されたDO変数の範囲の中で、DO変数の最終値だけを変化させ、DOループの繰返し回数の増減を試みる。

２．３ 測定項目

以下の４項目に関してデータを収集する。

(1) ループ全体の実行時間

DOループ内の計算量の大小、および、DOスパンの増減によるセルフスケジューリング、プリスケジューリングの優劣を測定する。

(2) ループ中のビジー・ウェイト数

Testset-Test命令は、間隔の大きい繰返し間の同期に対して不利である。また、Send-Wait命令は、プリスケジューリングの場合、同期先の繰返しを実行するプロセスを直ちに検索できるが、セルフスケジューリングの場合、同期先プロセスを検索するシークエンスが必要である。ビジー・ウェイト、および、Fetch and Addによるスピン・ロックの回数を測定し、これらのオーバヘッドを把握する。

(3) 最初の繰返しにおけるビジー・ウェイト数

最初の繰返しにおけるFetch and Add、または、同期によって、各プロセッサの繰返しの実行時間が適当にずれることにより、２回目以降の繰返しでは、ビジー・ウェイトによるオーバヘッドが減少する。その減少の程度を測定する。

(4) ループ全体のキャッシュ性能

DOスパンの増大によってループ内のデータのリプレイスが発生することが予想される。そこで、DOスパンの増減、ループの種類によるキャッシュ性能の変化を測定する。

３．ＣＦＯＲの実現

上述の解析方針にもとづき、TOP-1上でCFORを実現している。CFORは、Cソース・プログラムのプリプロセッサである。入力したプログラム中の特殊並列キーワードを含むループを並列化し、かつ、オーバヘッド測定ルーチンを付加する機能を持つ。CFORに入力するオプションによって、並列ループの実行に変化を与えることができる。現在、CFORは、上述の機

Implementation of Parallel Loop Analyzing Tool: CFOR
Munehiro FUKUDA
IBM Research, Tokyo Research Laboratory, IBM Japan, Ltd.

能のうち、DOスパン制御、キャッシュ性能測定機能を除く、すべての機能を提供できる。

CFORに入力するプログラム例と並列実行の流れを図１に示す。CFORとECFORキーワードで囲まれた各ループを親子すべてのプロセスが同期を取り合いながら実行を行う。ループとループの間では、子プロセスは親プロセスを待ち合わせる。

子プロセスの生成、消滅、共有変数の獲得、解放は、システム・コールを利用する。プロセス間の待合わせは、共有変数へのビジー・ウェイトにより、また、Fetch and Addは、スピン・ロックにより実現する。

SEND-WAIT命令は、各プロセスがSCB(Synchronization Control Block)とINS(Iteration Number Storage)の２つの共有領域を互いに参照することにより実現される。SCBは、プロセスごとに割り当てられ、プロセスが実行している繰返し番号、SEND命令の発行回数を保持する。INSは、すでに処理された繰返し番号列を保持する。TESTSET-TEST命令は、各繰返しがICB(Inter-Iteration Communication Buffer)に繰返し番号を順に書き換えることにより実現される。

キャッシュ性能の測定は、TOP-1のキャッシュ・メモリに内蔵されている統計機能、および、この機能を制御するパッケージ[5]を利用する。

プログラム例

```
.....
cfor (i = 2; i < N; i+= 1) {
        a[i] = b[i] + c[i];
        send();
        wait(1, -2);
        d[i] = e[i] + a[i-2];
    } ecfor;
.....
```

プログラムの流れ

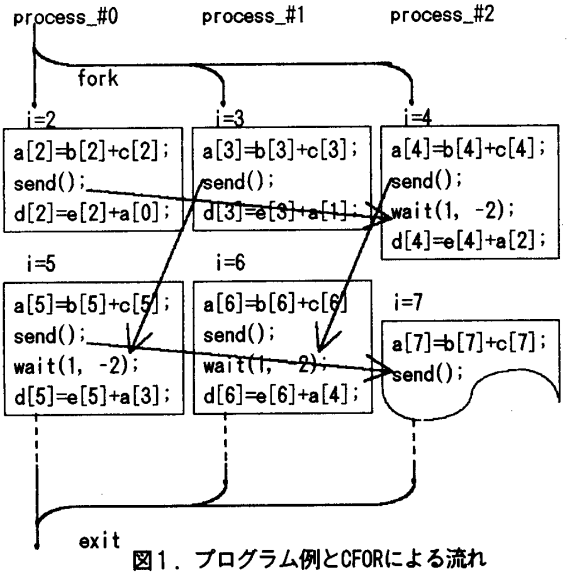

図１．プログラム例とCFORによる流れ

４．リバモア・ループの並列実行

リバモア・ループのCソース・プログラム中の各ループをCFOR、ECFORキーワードで置き換え、データ依存がある個所にSEND-WAIT同期命令を挿入し、TOP-1上で並列実行する。図２、および、表１にCFORによるリバモア・ループの並列実行の結果を示す。数行程度の文からなるループに対しては、プリスケジューリングが有効であることが裏付けられる。セルフスケジューリングは、ビジー・ウェイト数がプリスケジューリングの４分の１以下であるにもかかわらず、良好な性能を得られない。これは、セルフスケジューリングにおけるSEND-WAIT命令が、同期先プロセスの検索に必要な実行シークエンスの影響を強く受けていることによる。

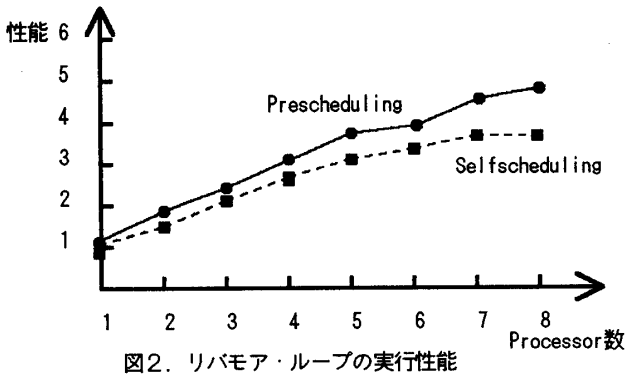

図２．リバモア・ループの実行性能
(核ループ#1 - #14, 但し、#4 および #11を除く)

表１．核ループ#13をプロセッサ８台で実行したときのプロセッサ１台当たりのbusy wait数、spin lock数　(平均)

	Preschedule	Selfschedule
繰返し実行回数(A)	64	64
busy wait総数(B)	1362.6	332.3
繰返し1回当たりのbusy wait数(B/A)	21.3	5.2
最初の繰返しにおけるbusy wait数	465.6	82.9
実行性能(倍)	6.3	5.4

５．むすび

並列ループ解析ツールCFORの実現について述べた。今後は、CFORの機能を充実させるとともに、より多くの演算パッケージを CFORを用いて並列実行し、オーバヘッド解析、並列支援機構の検討に利用する予定である。

参考文献

[1]N. Ohba, A. Moriwaki, S. Shimizu: TOP-1: A Snoop-Cache-Based Multiprocessor, Proc. IPCCC, March 1990

[2]F. MacMahon: The Livermore Fortran Kernels: A Computer Test of the Numerical Performance Range, Lawrence Livermore National Laboratory Report UCRL-53745, 1986

[3]M. Wolfe: Multiprocessor Synchronization for Concurrent Loops, IEEE Software, Jan. 1988, pp. 34-42

[4]S. P. Midkiff, D. A. Padua: Compiler Generated Synchronization for Do Loops, Proc. ICPP, CS Press, 1986, pp. 544-551

[5]大庭, 小原, 山崎, 清水: 並列処理ワークステーション TOP-1の性能評価環境, SWoPP琉球'90, July 1990