

山名早人 村岡洋一
早稲田大学理工学部

k個のフロー依存を持つDOACROSS型ループ実行時のデータ通信発行遅延時間TD_i及びデータ通信発行時刻TT_iを求めるアルゴリズム.

```

INPUT : R, TM, Ss(i) (i=1,...,k)
OUTPUT : TD, TT (i=1,...,k)
TDi = 0
TTi = T(S1, Ss(1))
for i=2,k
    PITCH=T(S1, Ss(i))-TM
    ;Qi-1のデータ通信発行時刻からCiのデータ定義までの時間
    if (PITCH+TM<Pc) then
        TD=Pc-PITCH-TM
        TT=T(S1, Ss(i))+TM+TDi
    else
        TD=0
        TT=T(S1, Ss(i))+MAX(0, PITCH)
    endif
endfor

```

図4 データ通信発行遅延とデータ通信発行時刻を求めるアルゴリズム

図5において、各文の実行時間を全て1単位時間、 $P_c=2$ とすると、図4の手順により、 $TD_1=0$, $TD_2=0$, $TD_3=1$, $TD_4=0$ となる。これは、 P_c の間隔でデータ通信を行った時、 C_3 のデータ通信が1単位時間遅延することを示す。

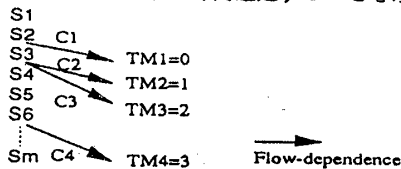


図5 データ通信発行遅延TD_iの算出例

3.4 通信ピッチを保証するディレイ d_p

データ通信発行時及びデータ受信時に通信ピッチ P_c の間隔でデータ通信が行われる時のディレイ d_p を求める。

3.4.1 発行間隔保証 TD_i によってDOACROSS型ループが各イテレーション毎に遅れる時間は、 TD_i/Δ_i なので、 TD_i/Δ_i ($i=1,\dots,k$)の最大値で示される時間分、各イテレーションの実行開始が遅延を受ける。

$$d' = d_0 + \max_{i=1,\dots,k} (TD_i / \Delta_i) \quad (2)$$

3.4.2 受信間隔保証 データ通信発行が通信ピッチ P_c の間隔で行われ、かつ、通信ピッチ P_c の間隔でデータが受信される時のディレイ d_p を求める。ディレイ d' で各イテレーションが実行される時、フロー依存によるデータ通信は、各イテレーションの開始時刻を0とすると、時刻 TT_i に発行される。時刻 TT_i に発行されたデータは、データ通信遅延時間 δ 後、ディスティネーションPEに到着する。ディスティネーションPEは、ソースPEに比較して、 $\Delta_i \times d'$ 時間遅れて実行を開始しているので、ディスティネーションPEでは、時刻 $TT_i + \delta - \Delta_i \times d'$ (データ到着時刻 RT_i で表す)に到着する。しかし、ディスティネーションPEは、通信ピッチ P_c より短い間隔でデータを受け取れないため、送信されたデータは相互結合網中に一時的にバッファリングされ、受信時刻が遅延する場合がある。この遅延を考慮したデータ受信時刻 (RT_i で表す) をk個のフロー依存によるデータ通信について求め、受信データが演算中で参照される時刻 ($REFT_i$ で表す) までに全て受信されるようにディレイ d_p ($\geq d'$) を求める。求出手順は、本報告では割愛する。詳細は文献^[4]を参照していただきたい。

3.5 DOACROSS型ループ実行時間

通信ピッチ P_c を考慮しない場合、すなわち、 P_c が十分に小さく、データ通信の発生間隔でデータ通信が出来る場合と、 P_c の間隔でデータ通信が行われるとした場合の各々について、DOACROSS型ループの実行時間を求める。

(1)通信ピッチ P_c を考慮しない場合 文献^[1]に示される実行時間と等しい。実行時間を T_0 とすると次式となる。

$$T_0 = d_0 \times (N-1) + T(S_1, S_m) \quad (3)$$

(2)通信ピッチ P_c を考慮する場合 P_c の間隔でデータ通信が行われる時の実行時間 T_p は、次式となる。

$$T_p = d_p \times (N-1) + T(S_1, S_m) \quad (4)$$

以上より、通信による実行時間遅延 T_d は、

$$T_d = T_0 - T_p = (d_0 - d_p) \times (N-1) \quad (5)$$

となる。このように、通信ピッチ P_c を考慮することによって、実行時間に遅延が生じることが明らかになる。

4. 最適データ通信順序

本節では、データ通信順序を変更し、ディレイ d_p をディレイ d_0 に近付ける手法を提案する。

4.1 データ通信順序が実行時間に与える影響

図2の例では、 $d_0=2$ 、フロー依存 C_1, C_2, C_3 のデータ通信余裕時間 TM_i は、2,3,0単位時間となる。データ定義順でデータ通信を行った時、 $P_c=2$ とすると、データ通信発行遅延時間 TD_i は、0,0,2となる。この時、通信ピッチを保証するディレイ d_p を求めると $d_p=4$ となる。これに対して、 $[C_1, C_2, C_3]$ の順で通信すると、全てのデータ通信発行遅延時間を0にできる。この時、通信ピッチを保証するディレイ d_p を求めると $d_p=2$ となる。データ通信順序を変更する前後での実行時間の差は、式(5)より、 $T_d = (4-2) \times (N-1)$ となり、データ通信順序を変更する前の実行時間が $(4 \times (N-1) + T(S_1, S_m))$ であるので、Nが十分に大きい時、ほぼ2倍に高速化される。

4.2 最適データ通信順序求出

DOACROSS型ループ内にk個のフロー依存が存在する時、k!通りのデータ通信順序が存在する。このk!通りの順序中には、データ定義順でデータ通信する場合のディレイ d_0 と同一、あるいは、ディレイ d_p より大きなディレイを持つ順序対が存在する。このような順序対は、データ通信発行時に生じるデータ通信発行遅延時間に着目すると、 $d_p \leq d_0 + \max(TD_i / \Delta_i)$ となる TD_i を持つ順序対である。このような順序対をk!通りのデータ通信順序対から除いた順序対についての、 P_c を考慮する場合の実行時間遅延 T_d を求め、 T_d を最小とするデータ通信順序を求める。求出手順は、本報告では割愛する。詳細は文献^[4]を参照していただきたい。

図2の例では、3つのデータ通信が存在し、全ての順序の組み合わせは3!=6通りとなるが、本節で示した手順を用いることにより、 $[C_1, C_2, C_3]$ と $[C_2, C_3, C_1]$ の2通りの順序対を得ることができる。この2通りの場合について、ディレイ d_p を求め、式(5)で与えられる実行時間遅延 T_d を最小にする順序を求め、最適なデータ通信順序とする。図2の例では、 $[C_1, C_2, C_3]$ の順が最適なデータ通信順序となる。

5. むすび

本報告では、DOACROSS型ループ実行時における最適なデータ通信順序を求める手法を示した。本手法は、(データ通信発生時間間隔)<(通信ピッチ)の場合に、有効な手法である。最近の並列処理計算機は、PE間の通信能力に比較して、PE内部の処理能力が高くなる傾向にあり、本稿で示した手法は、今後、重要となると考えられる。

文献

- [1] Ron Cytron: "Doacross: Beyond Vectorization for Multiprocessors", Proc. of Int. Conf. on Parallel Processing '86, pp.836-844(1986).
- [2] Peiyi Tang, Pen-Chung Yew, and Chuan-Qi Zhu: "Compiler Techniques for Data Synchronization in Nested Parallel Loops", Proc. of ACM 1990 Int. Conf. on Supercomputing, pp.177-186(1990).
- [3] Hong-Men Su and Pen-Chung Yew: "Efficient Doacross on Distributed Shared-memory Multiprocessors", Proc. of Supercomputing'91, pp.842-853(1991).
- [4] 山名, 村岡: "DOACROSS型ループにおける最適なデータ通信順序", 電子情報通信学会論文誌分冊D投稿中