1N-1

ホップフィールドの神経回路網モデルを用いたフローグラフのデータフロー・マルチプロセッサへの最適割り当ての一手法

中島重義* 佐藤清次** 山下一美*

(* 大阪市立大学 ** 産業技術短期大学)

1. 序論

データフローマシンは細粒度な並列計算機の代表とされている。すなわち、演算レベルでの並列性を自動的に見いだして実行するのである。しかし、データフローマシンも、単体での処理能力には限界があり、最近ではデータフロー・マルチプロセッサの開発がいくつかの場所でなされている。データフロー・マルチプロセッサは、複数のデータフロー計算機がネットワークにより結合され、マルチプロセッサを構成しているものである。一般に、フローグラフを静的に解析した結果をマルチプロセッサに最適に割り当てる問題はNP困難になることが知られており、制限を用いるなどして多項式のオーダーで解く方法が提案されている[1][2][3]。

一方、組合せ最適化問題に関しては神経回路網モデルによって解く方法が提唱されており、武田らはHopfieldのモデルを用いてHitchcock問題を解く方法について述べている[4]。

本論文ではこのデータフロー・マルチプロセッサに対しフローグラフを静的に割り当てる問題において、Hopfieldのモデル[5]を用いて多項式のオーダーで解を得る方法を提案する。

2. フローグラフ割り当てにおける問題点

データフロー型計算機では、演算の並列実行が動的になされる。しかし、フローグラフが処理エレメント、つまりデータフロー・マルチプロセッサを構成する個々のデータフロー型計算機にまたがって存在する時には動的、または静的に割り当てなければならない。動的な割り当てはオーバーヘッドが大きい。静的な割り当ては、ノードの実行時間の動的な変化に対応できないが、そうでない場合は動的な割り当てよりも実行時のオーバーヘッドが少ない。ここでは静的な割り当てについて述べる。この割り当てにおける評価基準にはフローグラフ全体の実行に要する時間の遅れを考えるのが妥当であろう。フローグラフ全体の実行時間に影響するものとしては次のような項目が考えられる。

- 個々の処理エレメントでの並列実行による遅れ
- 処理エレメント間のトークン授受による遅れ

3. コスト関数

前章の議論で、フローグラフ全体の実行時間への影響を考えて割り当てることについて述べたが、ではこの遅れを実際にはどの様にして静的解析から求めれば良いかについて考える。最も厳密に考えれば、それらの時間遅れも考慮したフローグラフの最長パスを求めることが必要である。しかし静的解析からの最長パスの算出は割り当ての変化によって再計算が要求される点から、困難が予想される。

最長パスを計算しない場合は、個々の部分の時間遅れが全体の時間遅れに影響する確率を考え、それらの総和を取る方法が考えられる。

静的解析に於て、フローグラフの中で並列に実行されるノードを抽出するためには、ランク解析が用いられる。ランク解析とは、ノードのランクをルートノードからの最短距離であるとし、同一ランクのノードは同一のタイミングで実行されるものとして解釈する。各ランクのノードを処理エレメントに割り当てる事による時間遅れが問題になる。まず個々の処理エレメントにおける並列実行による遅れについて考える。これは各ランク各処理エレメントで実行されるノード数に依存する。データフロー計算機の上で同時に実行可能となるノードは、ハッシュ関数によってハッシングされている場合がある[6]。この時、ハッシュ関数の衝突が起き、これが実行時間の遅れに影響する。あるノードと他のノードの間にハッシュの衝突が起きる確率が一定であると仮定すると、ある処理エレメントで同時に実行されるノードの間のペアの数、つまりノード数がmならば、m(m−1)がハッシュ関数の衝突回数の期待値に比例する。ここでは、並列実行の影響はm(m−1)に比例するものと考える。

処理エレメント間のネットワークの上でトークンの授受が行われると、トークンがネットワークに送られない状態に比べて、トークンを受け取るノードの実行が遅れる。

以上のような考えに基づいてコスト関数Fを導入する。Fの値はフローグラフの実行時間の遅れを意味し、二つの項からなる。第一の項は処理エレメントにまたがるトークン授受によるフローグラフの実行時間の遅れを示す。もう一つの項はノードの並列実行によるフローグラフの実行時間の遅れを示す。まず第一の項について述べる。

ノードn, n'とランクrに対し次のRとLを定義する。

$$L(n,n')=\begin{cases}1 & \text{nからn'までリンクが存在する場合}\\ 0 & \text{そうでない場合}\end{cases} \quad (1)$$

$$R(n,r)=\begin{cases}1 & \text{nがランクrに存在する場合}\\ 0 & \text{そうでない場合}\end{cases} \quad (2)$$

さらにノードnがプロセッサpに割り当てられている確率をV_{np}で示す。割り当てが確定したとき次が成立する。

$$V_{np}=\begin{cases}1\\ 0\end{cases} \quad (3)$$

$$\sum_{p} V_{np}=1 \quad (4)$$

あるノードnからn'へのリンクが存在していて、nがpに、n'がp'にそれぞれ割り当てられることによる通信量の期待値が$c_{nn'pp'}$であったとする。次が成り立つ。

A Method of Optimal Allocation of Flow Graphs to Data Flow Multiprocessors with Hopfield's Neural Network Model

Shigeyoshi Nakajima*, Kiyotsugu Satoh**, Kazumi Yamashita*
*Osaka City Univ. **College of Industrial Technology

$$c_{nn'pp'} = (1-\delta_{pp'})\, V_{np} V_{n'p'} \quad (5)$$

ここで$\delta_{pp'}$はクロネッカのデルタ関数である。

$c_{nn'pp'}$の単位は、一つのトークン授受に要する時間とする。一般的に、V_{np}と$V_{n'p'}$によるネットワークの通信量$c_{nn'pp'}$は次のようになる。

$$c_{nn'pp'} = L(n,n')(1-\delta_{pp'})\, V_{np} V_{n'p'} \quad (6)$$

ノードnとノードn'によるネットワークの通信量$c_{nn'}$は次のようになる。

$$c_{nn'} = \sum_{p}\sum_{p'} L(n,n')(1-\delta_{pp'})\, V_{np} V_{n'p'} \quad (7)$$

よって、システム全体の通信量cは次のとうりになる。

$$c = \sum_{n}\sum_{n'}\sum_{p}\sum_{p'} L(n,n')(1-\delta_{pp'})\, V_{np} V_{n'p'} \quad (8)$$

次に、第二の項について述べる。ノードnとn'が同一のランクrにあるとき、これが同一のプロセッサpにある確率は次のようになる。

$$V_{np} V_{n'p} \quad (9)$$

よってノードnとn'が同一のランクrにあるとき、これが同一のプロセッサにある確率は次のようになる。

$$\sum_{p} V_{np} V_{n'p} \quad (10)$$

次のように書いてもよい。

$$\sum_{p}\sum_{p'} \delta_{pp'} V_{np} V_{n'p'} \quad (11)$$

一般にノードnとノードn'が同一ランクで同一プロセッサに割り当てられる確率は次のとうりである。

$$\sum_{r}\sum_{p}\sum_{p'} R(n,r)R(n',r)\,\delta_{pp'} V_{np} V_{n'p'} \quad (12)$$

あるノードnから同一ランクで同一プロセッサに割り当てられている別のノードに有向な並列性リンクを引く。その本数の期待値は次のような値を取る。

$$\sum_{n'} (1-\delta_{nn'}) \sum_{p}\sum_{p'}\sum_{r} R(n,r)R(n',r)\,\delta_{pp'} V_{np} V_{n'p'} \quad (13)$$

よってプログラム全体では、この並列性リンクの本数の期待値は次のようになる。

$$\sum_{n}\sum_{n'} (1-\delta_{nn'}) \sum_{p}\sum_{p'}\sum_{r} R(n,r)R(n',r) \cdot \delta_{pp'} V_{np} V_{n'p'} \quad (14)$$

割り当てが確定したとき、この式の値は個々のプロセッサの個々のランクに置ける$m(m-1)$を全体で総和したものになる。

以上の議論によって、コスト関数を次のように設定する。

$$F = C\sum_{n}\sum_{n'}\sum_{p}\sum_{p'} L(n,n')(1-\delta_{pp'})\, V_{np} V_{n'p'} + C'\sum_{n'}\sum_{n}\sum_{p}\sum_{p'}\sum_{r} (1-\delta_{nn'}) \cdot R(n,r)R(n',r)\,\delta_{pp'} V_{np} V_{n'p'} \quad (15)$$

CとC'はそれぞれ処理エレメントにまたがるトークン授受とノードの並列実行が全体の処理時間の遅れに及ぼす影響の大きさを示す係数である。

4. Hopfieldの神経回路網モデルによる解決法

Hopfieldの神経回路網モデルはi番目のニューロンの状態V_iに関し(Tij)が対称行列であるかぎり、次の形で示されるエネルギー関数Eの極小値を求めることができるとされる[5]。

$$E = -(1/2)\sum_{ij} T_{ij} V_i V_j - \sum_{i} I_i V_i \quad (16)$$

この性質を用いて条件式（3）（4）を満足し、かつコスト関数Fを最小とする解を求めれば良い。ここでは（16）式の代わりに次を用いる。

$$E = -(1/2)\sum_{npn'p'} T_{npn'p'} V_{np} V_{n'p'} - \sum_{np} I_{np} V_{np} \quad (17)$$

また条件式（3）を満足するように次の項を加える。

$$-\sum_{n}\sum_{p} (1-2V_{np})^2 = -NP + 4\sum_{n}\sum_{p} V_{np} - 4\sum_{n}\sum_{p}\sum_{n'}\sum_{p'} \delta_{nn'}\,\delta_{pp'} V_{np} V_{n'p'} \quad (18)$$

さらに条件式（4）を満足するために次の項を加える。

$$\sum_{n} (1-\sum_{p} V_{np})^2 = N - 2\sum_{n}\sum_{p} V_{np} + \sum_{n}\sum_{n'}\sum_{p}\sum_{p'} \delta_{nn'} V_{np} V_{n'p'} \quad (19)$$

よって、Eは次のように設定すればよい。

$$\begin{aligned} E = {} & A\{4\sum_{n}\sum_{p} V_{np} - 4\sum_{n}\sum_{p}\sum_{n'}\sum_{p'} \delta_{nn'}\,\delta_{pp'} V_{np} V_{n'p'}\} \\ & + B\{2\sum_{n}\sum_{p} V_{np} + \sum_{n}\sum_{n'}\sum_{p}\sum_{p'} \delta_{nn'} V_{np} V_{n'p'}\} \\ & + C\sum_{n}\sum_{p}\sum_{n'}\sum_{p'} L(n,n')(1-\delta_{pp'})\, V_{np} V_{n'p'} \\ & + D\sum_{n}\sum_{p}\sum_{n'}\sum_{p'}\sum_{r} (1-\delta_{nn'})\, R(n,r)R(n',r) \cdot \delta_{pp'} V_{np} V_{n'p'} \end{aligned} \quad (20)$$

よって$T_{npn'p'}$とI_{np}に関しては次の式が成立する。

$$\begin{aligned} T_{npn'p'} = {} & 8A\,\delta_{nn'}\,\delta_{pp'} - 2B\,\delta_{nn'} \\ & - 2CL(n,n')(1-\delta_{pp'}) \\ & - 2D\sum_{r} (1-\delta_{nn'})\, R(n,r)R(n',r)\,\delta_{pp'} \end{aligned} \quad (21)$$

$$I_{np} = -4A - 2B \quad (22)$$

5. 結論と今後の方向

Hopfieldのモデルがフローグラフの静的割付に応用できることが示された。しかし、このモデルは漸近的な変化を利用するので、局所最小値におちいる可能性がある。また、拘束条件がエネルギー関数に含まれているので、最終的な解が拘束条件を満足しない場合もある。今後はエネルギー関数の係数の値の影響を調べ、この手法の有効性を示すようにしたい。

参考文献

[1] M.L.Campbell, "Static Allocation for a Data Flow Multiprocessor," Proc. Int. Conf. Parallel Process, pp. 511-517, 1985.

[2] H.Kasahara and S.Narita, "Practical Scheduling Algorithms for Efficient Parallel Processing," IEEE Trans. Comp., 33, pp. 1023-1029, Nov. 1984.

[3] S.Bokhari, "Partitioning Problems in Parallel, Pipelined, and Distributed Computing," IEEE Trans. Comp., 37, pp. 48-57, Jan. 1988.

[4] M.Takeda and J.W.Goodman, "Neural Networks for Computation: Number Representations and Programing Complexity," Applied Optics, 25, pp. 3033-3046, Sep. 1986.

[5] J.J.Hopfield and D.W.Tank, "Neural Computation of Decisions in Optimization Problems," Biological Cybernetics, 52, pp. 141-152, 1985.

[6] 山崎哲夫 他,「外部ハッシュキーを用いたデータ駆動形実行制御方式の性能評価」, データフローワークショップ１９８７予稿集, pp. 287-294, 1987.