8L−2

分散メモリ型並列計算機における手続き型言語の実行方式*

野村 光紀　村越 英樹　舟久保 登†

東京都立科学技術大学‡

1　はじめに

共有メモリのボトルネックを解消する方式として分散メモリ・メッセージパッシング(DMMP)型並列計算機が注目されている．しかしデータの格納方法やメッセージの受け渡しをプログラマが管理しなければならないため，従来の手続き型言語のように手続きの実行時刻や依存関係を意識せずにこれを用いることは困難である．

一方，手続き型言語で与えられたプログラムをフロー解析することによってマクロタスク間のデータやコントロールの依存関係を抽出する手法は既に確立している[1]．

そこで本論文では，マクロタスクを単位とした分配をおこなわずアクタによって操作を受ける個々の変数を主体とした分配をおこなうことによって，分散メモリ型並列計算機上でユーザが個々の手続きの実行について意識することなしに効率の良い並列処理の環境を実現することを試みた．

2　プログラムのコンパイルと実行

まず対象となる手続き型言語をコンパイルする際に，プロセッサへの割り当ての単位となる変数の最適な抽出[2]とそれらに作用するアクタの列挙，およびそれらの変数のデータ依存，制御依存関係の解析をおこなう．

本方式では共有メモリを持たない並列計算機を想定している．それぞれローカルメモリを持ったプロセッサ一つ一つをノードとし，実行の際には，一つのデータとそれに関わるアクタを一つのノードに置く．アクタは他のノードからのメッセージを受けて，自分以外のデータが必要な場合そのデータを要求するメッセージを送信し，データが揃ったら実行をおこない，処理が終わったことを示すメッセージを送信するという一連の処理をおこなう．

コントロール依存の制御のために，変数を持たないノードも存在する．一つはコントロールノードで，手続きごとに作られ関数内での変数の生成やコントロール依存の制御をおこなう．もう一つは配列アクセスノードで，配列へのアクセスやループの制御をおこなう．

これらの動作の例を，図1に示す．

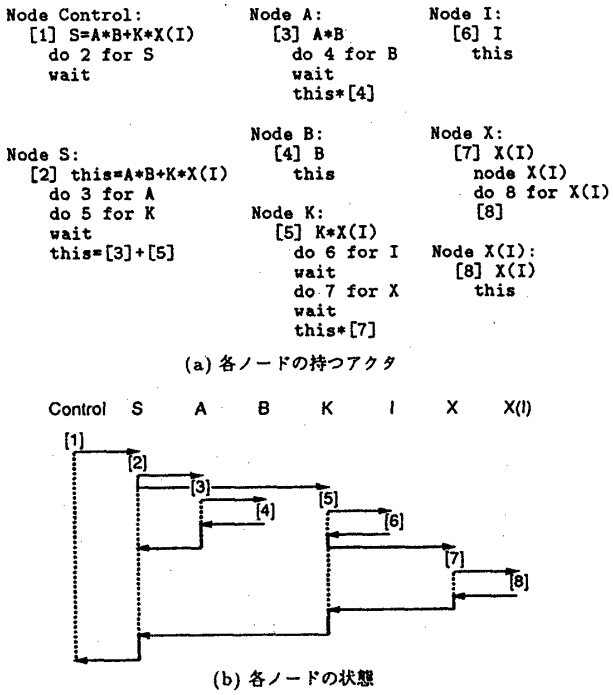


図 1: S=A*B+K*X(I) の実行例

3　制御構造の展開とメッセージ処理

手続き型言語には種々の制御構文が存在する．これらをデータフロー解析してノードプロセッサ上に展開する手法を次に挙げる．

条件分岐 コントロール依存の制御をおこなうノードは，まず条件式の評価をおこなうノードに対してメッセージを送り，評価の結果であるメッセージを受け取るまで同期待ち合わせをおこなう．その結果に応じて次のノードにメッセージを送る．

ループ ループは並列実行できる部分とできない部分に分けられる．並列実行できる部分に対しては，操作を受ける変数に対して非同期にメッセージを送る．ループの実行に依存する部分に対しては，依存する変数に対してメッセージを送り，その変数の処理が終りしだいその変数に依存する変数に対して順次メッセージを送る．配列アクセスノードは他のノードに影響を与えるデータを持たないので，待ち合わせの

---

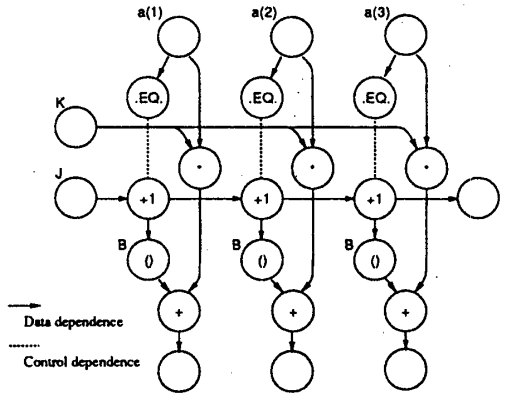
*An Approach to Computing for Procedural Language on Distributed Memory Multiprocessor

†Mitsunori NOMURA (E-mail: nomura@tmit.ac.jp), Hideki MURAKOSHI, Noboru FUNAKUBO

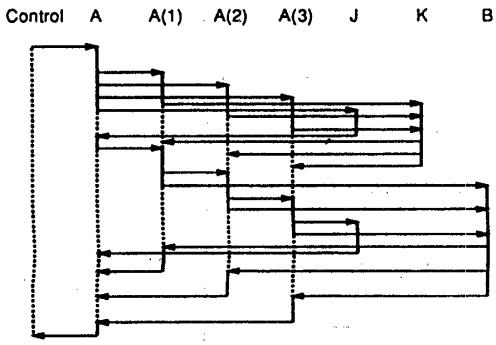
‡Tokyo Metropolitan Institute of Technology

```
DO I=1,10
  A(I)=K*A(I)+B(J)
  IF(A(I).GE.1) J=J+1
END DO
```

(a) ループを含むプログラム



(b) ループ展開後のフローグラフ



(c) 各ノードの状態

図 2: ループの実行例

途中でも別のメッセージを処理できる．これらの処理をコントロールノードがすべておこなうとボトルネックの原因となるので，配列アクセスノードがおこなう (図2).

手続き呼び出し コントロールノードは呼び出す手続きのための新しいコントロールノードを生成する．

4 実際の実行と評価

上で述べた実行方式では，ハードウェアやメッセージ交換の方法については特に規定しなかった．そこで，次のような制約をつけた簡単な方法でシミュレーションをおこなった．

コンテキストスイッチングはおこらないものとし，結合網はメッセージの交換のみに用いられる．メッセージ交換に必要な時間はノードの場所にかかわらずノードプロセッサの演算にかかる時間の 1/10 で，一つのノードが送信しているときは他のノードは送信できない．ノードの最大数は 100 である．

ループの一部分でも展開できる場合は待ち合わせの必要なメッセージのうち非同期の待ち合わせでよいものが 1/4 を越えた．これは逐時処理の必要なループでも効率的に処理できる可能性があることを示している．また，変数の操作が終了したことを示すコントロールメッセージは 1/3 程度であった．このメッセージは軽量であるが広範囲に渡るので，より密な専用の結合網を用いたほうがよいと考えられる．

5 実装時の諸問題に関する考察

通信方法 結合網の負荷を低くするため，結合網の方式が問題となる．手続き型言語の場合は一つの変数が使われるのは時間的にも空間的にも限られた範囲であることが多いので，ノードの配置方法が問題となる．また本方式では分散メモリ型並列計算機の弱点である短いメッセージを多用するため，低レイテンシのメッセージ通信が不可欠である．

コンテキストスイッチング コンテキストスイッチングのオーバヘッドを減少させる方法が問題となる．手続き型言語の場合は変数の寿命は比較的安定しているため，コンパイル時にメッセージのタイミングを変えることによって効率が上がる可能性がある．また，結合方式によって効果的なキャッシュが必要であろう．

先行評価 データの依存関係に影響を与えない場合，先行評価をおこなうことによって同期待ち合わせの時間を減らすことができる．しかし，先行評価のために必要なメッセージのために結合網の負荷を上げてしまうので，検討が必要である．

6 おわりに

本論文では従来のようにコードではなくデータを主体とした分配によって手続き型言語のプログラムを分散メモリ型並列計算機上で実行する方法を提案した．この方法では結合網でやりとりされるデータの量を減らすという点で効果がある．さらに簡単な例を用いて従来の手続き型言語でも分散メモリ型並列計算機で効率良く実行できることを確認した．

今後は実際に本方式を用いた処理系を構築し，実装時の問題について検討していく．

参考文献

[1] A. V. Aho, R. Sethi and J. D. Ullman, *Compilers Principles, Techniques and Tools*, Addison-Wesley (Mar. 1986).

[2] M. Gupta and P. Banerjee, *Demonstration of Automatic Data Partitioning Tequniques for Parallelizing Compilers on Multicomputers*, IEEE Trans. Parallel and Distributed Systems, 3, 2, pp.179-193 (Mar. 1992).