超オブジェクト並列プログラミング言語MAPPLEの実装

6J－2

福見 幸一　小野 剛　今崎 憲児　天野 浩文　牧之内 顕文
（九州大学 工学部 情報工学科）

1. はじめに

近年，プログラミングやデバッグが容易であるなどの理由から，様々なデータ並列言語が提案されている．データ並列言語はデータ構造として配列を利用しているため，数値計算などの応用分野では有効であると思われる．

しかし，データベースやCADなどのデータインテンシブな応用分野では，データ型の制限，データの静的配置などによる様々な問題が生じる．

本論文では，データインテンシブな応用分野で生じる問題点を指摘し，それらの問題点を解決すべく現在我々が開発している超オブジェクト並列プログラミング言語MAPPLE(旧称 INADA/MPP)の設計思想と実装方法を述べる．

2. データインテンシブな分野におけるデータ並列言語の問題点

データベース，CAE，CAD，CAMなどのデータインテンシブな応用分野では，複雑な構造のデータを大量にリアルタイムで処理する．従って，これらの応用分野でデータ並列言語を用いると，次のような問題が生じる．

- 配列の要素の型が基本的なデータ型に限定されているため，リストなどの集合型を扱うオブジェクト指向データベースではデータの表現が難しい．
- データの個数や実プロセッサへのマッピングがコンパイル時に決定するため，時間によって処理すべきデータの個数が変動する場合に対処しづらい．

現在我々が設計，開発している超オブジェクト並列プログラミング言語MAPPLEはこれらの問題を解決し，数値計算だけでなくデータベースやCADなどの分野でも利用できる言語を目指している．

Implementation of Massively Object-Parallel Programming Language MAPPLE
Koichi Fukumi, Tsuyoshi Ono, Kenji Imasaki, Hirofumi Amano and Akifumi Makinouchi
Department of Computer Science and Communication Engineering,Kyushu University

3. MAPPLEの設計思想

MAPPLEはC++を最小限拡張した超オブジェクト並列プログラミング言語である．オブジェクト並列はデータ並列を拡張した概念である．オブジェクト並列ではオブジェクト(データ構造とそれを操作するための関数(メソッド)を合わせ持ったもの)を並列計算機の各PE (Processing Element)に分散させ，メソッドを1つずつ同期を取りながら並列に実行する(図1参照)．

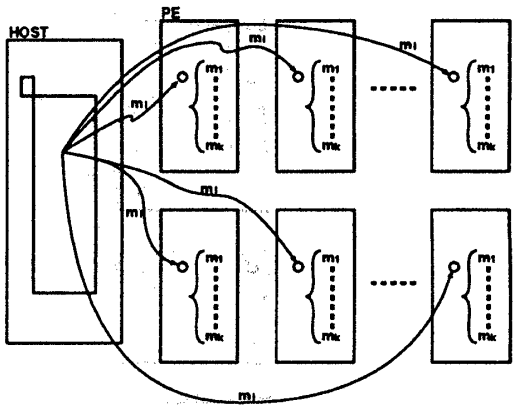

図 1: オブジェクト並列の概念図

MAPPLEではオブジェクトを集合オブジェクト(Set Object)とそうでない素オブジェクト(Atomic Object)に分ける．集合オブジェクトはデータの集まり(collection)であり，データ構造として“集合”のインターフェイスを備えていなければならない．集合オブジェクトの要素を要素オブジェクト(Element Object)と呼ぶ．要素オブジェクトは素オブジェクトでも集合オブジェクトでもよい．

また，MAPPLEではPE上のオブジェクトのメソッドを並列に実行するために，“和集合”という概念を導入している．和集合オブジェクト(Union Object)は各PE上の同一の型の集合オブジェクトまたは素オブジェクトを管理するオブジェクトである．従って，和集合オブジェクトは集合オブジェクトの特別なものと考えることができる．

これらのオブジェクトを模式的に表すと図2のようになる．

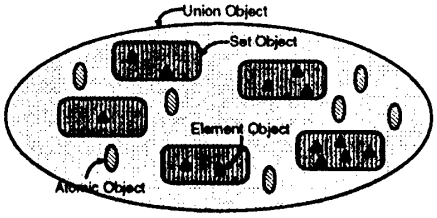

図 2: MAPPLEにおけるオブジェクト

4. MAPPLEの実装

図3はMAPPLEのコンパイル方式を示す. MAP-

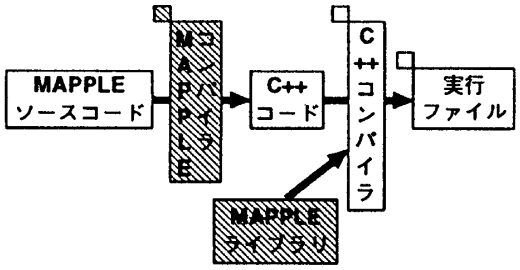

図 3: MAPPLEのコンパイル方式

PLEはC++の拡張言語であるため, MAPPLEコンパイラはC++のコードを出力する. そのC++のコードにMAPPLEのライブラリをリンクすることで実行ファイルが得られる.

以下にコンパイラ内部で行っている集合オブジェクトのクラス定義の書き換えとメッセージのまとめ送りによる最適化を説明する.

4.1 クラス定義の書き換え

MAPPLEでは和集合オブジェクトの要素オブジェクトおよびそれが集合オブジェクトの場合はその集合オブジェクトの要素オブジェクトのメソッドを並列に実行できる.

しかし, 集合オブジェクトの要素オブジェクト(以下, 単に要素オブジェクトと呼ぶ)は1つのPEに多数割り当てられており, 要素オブジェクト間で通信も可能である. 従って, 要素オブジェクトのメソッドを単に実行しただけでは正しい結果が得られない.

そこでMAPPLEコンパイラはあたかも要素オブジェクトのメソッドが並列に実行されているかのようにエミュレートするメソッドを集合オブジェクトのクラス定義に追加する. また, 要素オブジェクトのメソッド(m1())の並列実行は集合オブジェクトのエミュレーションメソッド(EMU_m1())の並列実行に書き換えられる(図4参照).

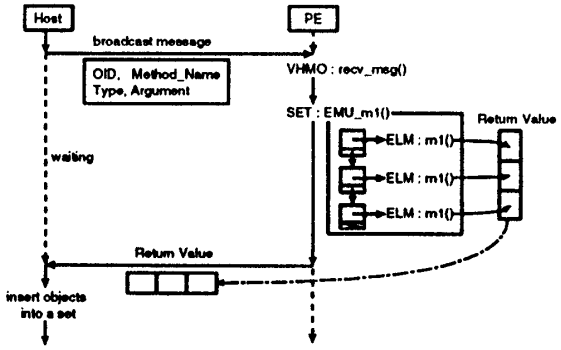

図 4: 要素オブジェクトのメソッドのエミュレート

4.2 メッセージのまとめ送り

並列処理では如何にしてPE間の通信を減らすかが効率を上げるポイントである. これに対しMAPPLEでは,

- ブロードキャストメッセージを多用する.
- メッセージをまとめる.

の2つの対策を講じている.

最近の並列計算機ではブロードキャスト専用の高速ネットワークが整備されているため, オブジェクト間の通信などメッセージの内容が異なる場合もPE内でできる限りメッセージをまとめ, ブロードキャストで送った方が効率が良い.

また, 並列に実行したメソッドが呼び出した側に戻り値を返す場合なども, 各PEで戻り値を1つにまとめ呼び出し側に送り返すことでメッセージの本数を減らしている(図4参照).

5. おわりに

本論文ではデータインテンシブな分野でデータ並列言語を用いる際に生じる問題点を指摘し, それらを解消するために現在我々が開発している超オブジェクト並列プログラミング言語MAPPLEの設計思想と実装方法について述べた. 我々は既に様々な例題に対する実装の予備的評価を終えており[AI94], 現在は上で述べたコンパイラを作成中である.

今後は永続データも扱えるように拡張し, データベースへの応用を検討していく予定である.

参考文献

[AI94] 天野, 今崎, 福見, 牧之内: "オブジェクト並列プログラミング言語INADA/MPPの設計方針と予備的評価について", 文部省重点領域研究第4回シンポジウム予稿集, 2-214 ～ 2-220, 平6.3.