

4AC-2

超並列プログラミング言語 MAPPLE のための
ヒープ管理オブジェクトの NOW への実装

堀渕 高照[†] 天野 浩文[‡] 牧之内 顕文[†]

[†]…九州大学大学院システム情報科学研究科知能システム学専攻

[‡]…九州大学大型計算機センター

1 はじめに

近年コンピュータ技術の発展により、プロセッサを数十台、数百台搭載した分散メモリ型並列計算機が商用化されている。分散メモリ型並列計算機は、各プロセッサがメモリを独自で持っているため、プロセッサの数を増やすのが容易で、全体で巨大なメモリと二次記憶を持つことができる。このため巨大なデータベースを構築することが可能になる。我々は、この考え方に基づいて超並列データベース MAPPLE/DB を開発している。超並列プログラミング言語 MAPPLE は MAPPLE/DB のデータベースプログラミング言語である。

これまでは、MAPPLE のプロトタイプを富士通社製並列計算機 AP1000 上で開発してきた。しかし AP1000 は二種類の通信路を持つ並列計算機であるため、その性能が他の並列計算機または NOW (Network of Workstations) でもそのまま達成できるとは考えにくい。そこで NOW へ MAPPLE を実装し性能の評価を行なうことが必要となる。

本論文では、MAPPLE 上のオブジェクトを管理するヒープ管理オブジェクトの、NOW 上での MPI[1] (Message-Passing Interface) による実装について述べる。

2 MAPPLE の概要

MAPPLE はデータ並列を拡張したオブジェクト並列の概念に基づき設計されている。オブジェクト並列とは、各プロセッサにオブジェクトを分散し、並列にメソッドを実行することである。したがって並列プログラミングが容易になり、また並列性を抽出することが容易になる。

オブジェクトは各プロセッサに分散されている。他のプロセッサにあるオブジェクトを参照するために、各オブジェクトは固有のオブジェクト識別子 (OID) を持つ。オブジェクトへの参照はすべて OID により行なわれる。MAPPLE ではヒープ管理オブジェクト

(HMO) を各プロセッサが持ちオブジェクトの参照による通信を少なくすることを実現している。HMO は OID を持つすべてのオブジェクトを管理する。(図1) ヒープ管理オブジェクトはオブジェクト参照テーブル (ORT) とオブジェクト参照テーブル (OAT) を持つ。ORT は、自分の PE に存在するオブジェクトの OID を持つ。オブジェクトの処理はすべて ORT を通して行なわれる。OAT は他の PE に存在するオブジェクトの OID とその PE 番号を持つ。

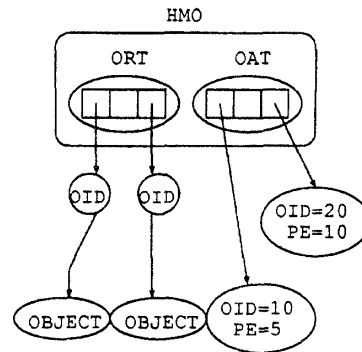


図 1: ヒープ管理オブジェクト

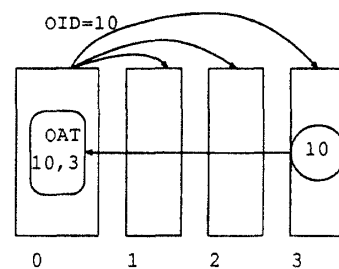


図 2: オブジェクト配置テーブル

3 オブジェクトへの参照

OID はプロセッサ番号を含まない。これはオブジェクトの移動により、各プロセッサの負荷の偏りを減らす機能を提供するからである。このため他のプロセッサのオブジェクトへの参照の時に、全てのプロセッサにブロードキャストをして、目的のオブジェクトが存在するプロセッサを見つけなければならない。しかしブロードキャストを参照のたびに使っていると、効率が悪くなる。そこで MAPPLE では、一度検索したオブジェクトの PE 番号を記録するオブジェクト配置テーブル (OAT) を用意している。(図2) OAT に

Implementation of Heap Management Object for a Massively Parallel Programming Language MAPPLE on NOW

Takaaki Hoributi[†], Hirofumi Amano[‡], Akifumi Makinouti[†]

[†]… Dept. Intelligent Systems, Kyushu University

[‡]… Computer Center, Kyushu University

よって、同じオブジェクトへの二回目からの参照が1対1通信でできる。これらの処理をヒープ管理オブジェクトが行なう。

4 ヒープ管理オブジェクトの性能

4.1 AP1000での性能

これまで、MAPPLEのプロトタイプを富士通社並列計算機 AP1000 上で開発してきた。ヒープ管理オブジェクトの OO1 ベンチマーク [2] によるトラバース処理の性能を図3に示す。要素となるオブジェクト数は10000個である。

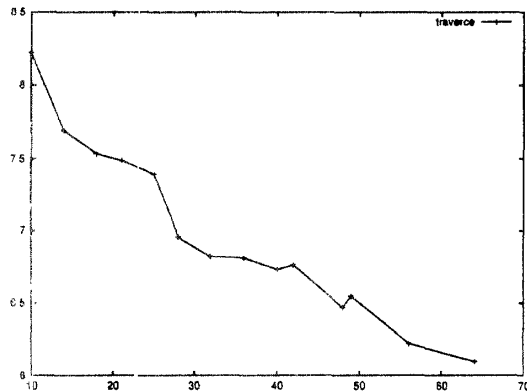


図3: AP-1000でのトラバース

AP1000には1対1通信とブロードキャストのそれぞれに別の通信路がある。またMAPPLEの参照方式はこの2種類の通信を使い分けている。このためこの結果がそのまま他の並列計算機またはNOWの場合にも当てはまるとは考えにくい。

4.2 NOWでの性能

NOWを仮想的な並列計算機として利用できるのがMPIである。ヒープ管理オブジェクトをMPIで実装しNOW上で実験を行なった。AP1000とMPIの通信方式の違いから、実装にも異なる部分がある。MPIの通信方式には1対1通信と集団通信がある。集団通信にはブロードキャスト、リダクションなどが含まれる。集団通信はメッセージを受け取る時に送信元のプロセス番号を指定しなければならない。このためプログラムを書く時点でブロードキャストの順番を固定しておかなければならない。ヒープ管理オブジェクトは処理の効率化のため、非同期に他のPEのオブジェクトを参照でき、また参照が必要な時にいつでも参照できるようにAP1000では実装している。MPIでも同じ機能を実現しなければならない。このため、今回の実装ではブロードキャストの代わりに1対1通信を繰り返すように実装した。今回の実験環境は1つの通信路で接続されているため、通信方式の違いによる処理時間の差は少ないと考えられる。

実験環境はLANで接続された8台のワークステーションである。図4にNOWでのトラバース処理の性能を示す。AP1000と同じく要素となるオブジェクト数は10000個である。

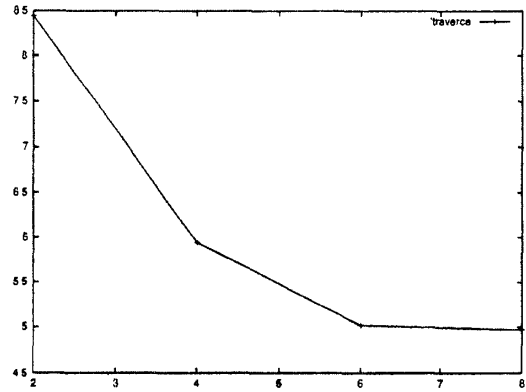


図4: NOWでのトラバース

5 評価

AP1000では64台まで処理時間が短くなり、並列性が認められる。しかしNOWでは6台と8台の差がほとんどなく、またこれ以上プロセッサを増やしても性能の向上は期待できない。プロセッサの処理速度に対して通信速度が遅いためこのような結果になったと考えられる。

NOWでの性能の向上のためには、更に通信回数を少なくしなければならない。

6 むすび

MPIを利用してのNOW上へヒープ管理オブジェクトの実装し、その評価を行なった。実験結果よりNOWでは更に通信回数を軽減しなければならないと考えられる。今後の課題としては、1回目の参照の後にOATを作るのではなく、オブジェクトが生成された時にOATを作る方式と比較を行なうことが考えられる。また、MPIが実装されている他の並列計算機で実験を行ない比較を行ないたい。

参考文献

- [1] Message Passing Interface Forum. "MPI: A Message-Passing Interface Standard," International Journal of Supercomputer Applications and High Performance Computing, 8(3/4), 1994.
- [2] Cattel, D. and Skeen, J.; "Object Operations Benchmark," ACM Trans. Database Syst., Vol. 17, No. 1, pp. 1-31, March 1992.