

分散共有オブジェクト

— 並列プログラミング・インタフェースの一検討 —

3F-6

山内 雅彦†, 吉澤 聡, 村山 秀樹, 林 剛久

(株)日立製作所 中央研究所

1 はじめに

近年, マルチプロセッサ WS(ワークステーション)の登場や複数の WS をネットワークで接続した分散環境の普及により, 並列処理を行なえる基盤が整いつつある。

並列計算機は共有メモリ型と分散メモリ型のアーキテクチャに大別することができる。共有メモリ型は, 従来の逐次計算機と同様なプログラミング・スタイルで使用できるが, スケーラビリティに問題がある。一方, 分散メモリ型は, スケーラビリティやアペイラビリティの面では優れているが, プログラミングが困難であると言われている。

上述の問題を軽減するため, 分散共有メモリ方式がいくつか提案されてきているが[1], メッセージ通信と比較すると並列処理による性能向上を得にくいという問題があった。

本報告では, 分散共有メモリのプログラミングの容易さを保ちつつ, 並列処理における性能向上を得られる API(Application Program Interface)として分散共有オブジェクトを提案する。筆者等は, 本提案の API を実現するために, 信頼性のあるブロードキャストを利用した一貫性保証方式を採用し, WS 上のプロトタイプシステムに実装した。本報告では, プロトタイプシステムによる方式の評価についても述べる。

2 分散共有オブジェクト

分散共有メモリによる並列プログラミングの容易さは, 次の点であると考える。

位置透過性: どのノードに最新データが存在するかをユーザが意識しない。

参照による共有: アドレスを指定することによっても共有データにアクセスすることができる。

一方で, 従来の分散共有メモリでは, 仮想記憶機構(ページング)を用いることが多いが, この方式には次の問題がある。

データ転送量: ページをデータ転送の単位としているため, ページサイズと比較して小さいデータを共有した場合には無駄が生じる。

データ移動回数: false sharing が発生した場合には, メモリ書き込み毎にページングが発生する可能性がある。

例えば false sharing を避けるためにページを意識する等, 分散メモリ型アーキテクチャ向けのチューニングが必要である。しかし, チューニングをするアプリケーションユーザにとっては, ページという物理的な単位を意識するのではなく, より論理的な単位(例えば変数レベル)で意識できるのが望ましい。

筆者等は, 上述の問題を解決するために, メッセージ通信の実行効率と分散共有メモリの利点を備えた分散共有オブジェクトを提案する。インタフェースの特徴は次の通りである(表1参照)。

- ユーザが共有データの粒度を意識し, オブジェクト生成時にデータサイズを宣言する。(dso_create)
- 共有オブジェクト番号とメモリ空間を対応付けて管理する。(表中のオブジェクト番号)
- ユーザが一貫性保証に関するタイミングを指定可能とする。
 - 共有オブジェクト獲得(dso_acquire)
 - 共有オブジェクト更新(dso_release)

表1. 分散共有オブジェクトのインタフェース

初期化处理	dso_create(オブジェクト番号, オブジェクトの開始アドレス, サイズ)
一貫性保証処理	dso_acquire(オブジェクト番号)
	dso_release(オブジェクト番号)
終了処理	dso_close(オブジェクト番号)

† 現在, (財)新世代コンピュータ技術開発機構出向中

Distributed Shared Object
— A Study of Parallel Programming Interface —
Masahiko YAMAUCHI, Satoshi YOSHIZAWA,
Hideki MURAYAMA, Takehisa HAYASHI
Hitachi, Ltd. Central Research Laboratory
1-280, Higashi-Koigakubo, Kokubunji-shi, Tokyo 185

3 システム実現方式

分散メモリ型計算機において共有オブジェクトを実現するためには、オブジェクトの一貫性保証方式をメッセージ通信によって行なう必要がある。また、メッセージ通信の方式には、さらに、1) ブロードキャスト使用、2) ユニキャスト (1 対 1 通信) のみの使用という選択肢がある。

前者の 1) では、データを必要としないノードに対して通信オーバーヘッドを与えてしまう。しかし、想定している台数規模に於ては、共有しているノード数に対して、一貫性保証の手続きを一定コストで実行でき、かつそのアルゴリズムが単純になるという利点がある。ただし、ここで用いるブロードキャストはイーサネットや FDDI が提供するブロードキャスト機能では不十分であり、信頼性を付加する必要がある¹ [2]。この信頼性のあるブロードキャストに基づいて、次の 2 通りのシステムのプロトタイピングを行なった。

システム 1：各ノードに共有オブジェクトの複製を保持する。共有オブジェクトを変更した時には、信頼性のあるブロードキャストでオブジェクトを共有しているノード上の複製を変更して一貫性を保つ (図 1 参照)。

システム 2：最新の共有オブジェクトは一つのノードで保持し、各ノード上では最新データを保持するノードの位置情報を管理する。共有オブジェクトを変更した時には、変更したノードの位置情報をオブジェクトを共有しているノードに伝達して一貫性を保つ。

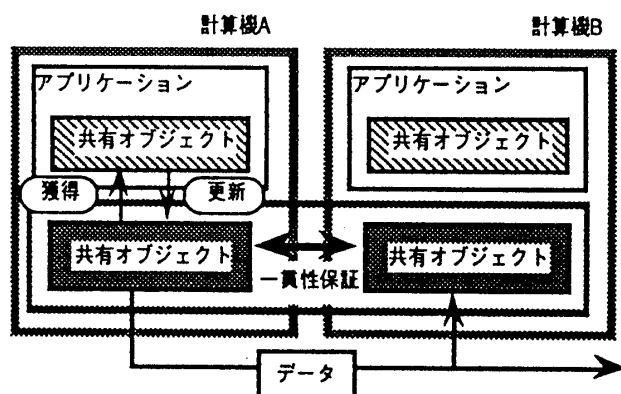


図 1: ブロードキャストを用いた実現方式

¹ここでいう「信頼性のあるブロードキャスト」とは、ブロードキャストしたパケットが消失することなく全ノードに到着し、かつ、ブロードキャストしたパケットを受信する順序が全ノードで必ず同じになることを保証する。

4 システム評価

システムの評価アプリケーションとして、2 次元 Poisson 方程式を Gauss-Seidel 法で解く問題を取り上げた。並列化の方針としては、2 次元メッシュ状のデータを等分割し、反復演算処理を分割メッシュ毎に異なるノードで並列に実行する。メッシュ分割した境界データは、分散共有オブジェクトを用いて共有する。

プロトタイプシステムの実験は、HP9000 700 シリーズ 4 台をイーサネットで接続した環境で行った。その結果を図 2 に示す。システム 1 では、図 2 で示すように、総データ数 200,000 程度以上の領域に於て、1 ノードで実行した場合より 3.6 倍以上の性能を得られることを確認した。

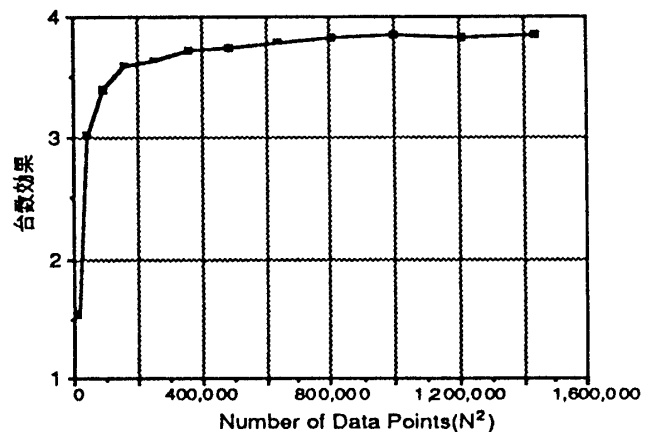


図 2: Gauss-Seidel 法への適用結果

5 まとめと今後の課題

分散メモリ型並列計算機に適した分散共有オブジェクト・インタフェースと、信頼性のあるブロードキャストを利用したプロトタイプ・システムについて述べた。今後、さらにインタフェースの記述性の確認と、プロトタイプ・システムによる性能評価を行ない、システムのチューニングにフィードバックしていく予定である。

参考文献

- [1] B. Notzner and V. Lo. "Distributed Shared Memory: A Survey of Issues and Algorithms," IEEE COMPUTER, 1991.
- [2] A. Tanenbaum, M. Kaashoek, H. Bal. "Parallel Programming Using Shared Objects and Broadcasting," IEEE COMPUTER, 1992.