ハイパフォーマンス
コンピューティング 79－6
（１９９９．１２．３）

JavaPM/Myrinet と SORB の性能評価

岡崎 史裕† 松田 元彦† 入口 浩一††

SORB は分散オブジェクト管理技術の抽象化されたインタフェースに集団実行と耐故障機能を実装することで、拡張性／耐故障性を持った並列サーバアプリケーションの開発を容易とする ORB である。JavaPM は SORB の通信レイヤであり、Java から PM/Myrinet による高速通信を利用可能とする。JavaPM はスレッド切替のオーバヘッドのためスループット 65MB/sec, レイテンシ 85usec の性能であった。SORB のリモート呼出しは Java RMI より 50%高速で、HORB と同程度の性能を達成した。

Implementation and Evaluation JavaPM/Myrinet and SORB

FUMIHIRO OKAZAKI †, MOTOHIKO MATSUDA †, HIROKAZU IRIGUCHI ††

SORB is an ORB (Object Request Broker) with aggregate method invocation and fault tolerance via redundancy, designed to ease development of parallel application servers. JavaPM is an adaptation layer of the existing communication layer PM/Myrinet in Java, designed to run primitives of SORB. JavaPM attains the communication performance of bandwidth 65MB/sec (throughput) and latency 85usec (round-trip). SORB attains the performance of remote method invocation 50% faster than Java RMI and is comparable to HORB's.

1．はじめに

多数のユーザに同時サービスを提供するサーバシステムには拡張性と高信頼性が必要とされる。クラスタ技術はこれを実現する技術として以前から利用されてきたが、プログラム設計の複雑さや価格が高価なことから中小規模のサーバシステムに適用されることは稀であった。ハード面では、プロセッサ性能の進歩により、コストパフォーマンスに優れた高性能な PC クラスタがサーバシステムとして注目されている。しかし、ソフト面では、障害時においてもサービスを継続できる耐故障性を備えた、並列アプリケーションを容易にプログラム設計可能なライブラリが必要である。

並列分散プログラムは、MPI、PVM などメッセージ通信ライブラリや RPC[1]、ORB などリモート呼出しで記述するのが主流である。これらライブラリには、サーバアプリケーションで必要とされる集団実行や耐故障の両機能をアプリケーションから透過的に記述できる機能がない。そこで我々は、ORB の抽象化されたインタフェースに集団実行と耐故障性を高めるための機能を実装す

† 新情報処理開発機構並列分散システム住友金属研究室
Parallel & Distributed System Sumitomo Metal Laboratory, Real World Computing Partnership
†† 新情報処理開発機構つくば研究センタ
Tsukuba Research Center, Real World Computing Partnership

るサーバアプリケーション構築用並列分散ライブラリ(SORB: Sumikin ORB)の開発を進めている。

SORB は、ORB のために必要なオブジェクトの直列化、リフレクションの機能を備えたオブジェクト指向言語 Java[3]で開発した。クラスタのプラットホームは、広く普及し、豊富なアプリケーションを持つ Windows NT とした。クラスタ通信環境は Myricom 社の Myrinet[3]と新情報処理開発機構つくば研究センタで開発された高速通信ライブラリ PM/Myrinet [4]を利用する。PM を Java のマルチスレッド環境で利用可能とする JavaPM を開発して、クラスタ内で SORB の通信レイヤに利用している。

本稿では、JavaPM と SORB の実装と性能評価を行ったので報告する。2 章では SORB のサーバシステムへの適用イメージについて説明し、SORB に必要な機能を明確にする。3 章では JavaPM の実装を、4 章では SORB の実装を説明して、5 章で性能評価を行う。最後に 6 章でまとめと今後の課題について述べる。

2. SORB の必要機能

SORB のサーバシステムへの適用イメージとして図 1 に示す情報検索サーバを例にとり、必要機能を説明する。

検索処理はクラスタによるデータパラレル型サーバとして N+1 台で冗長に構成し、検索要求を各ノードで集団実行した後、検索サーバで結果をソートして検索結果とする。インデックスは冗長構成で格納し、ノード故障時にはリモートオブジェクトを切替えてサービスを継続する。このようなサーバでは以下の 3 つの機能が必要と考える。

1) 集団実行機能
   1 回のリモート呼出しで複数ノードのリモートオブジェクトのメソッドを実行する機能。
2) マイグレーション機能
   障害時に同じ機能を持つ他ノードのリモートオブジェクトに切替える機能。保守などでノードを停止する場合にリモートオブジェクトを他ノードに移動する機能。
3) 高信頼オブジェクト管理機能
   内部状態を持ったオブジェクトをクラスタ内で冗長に持つことで、障害時に復元する機能。

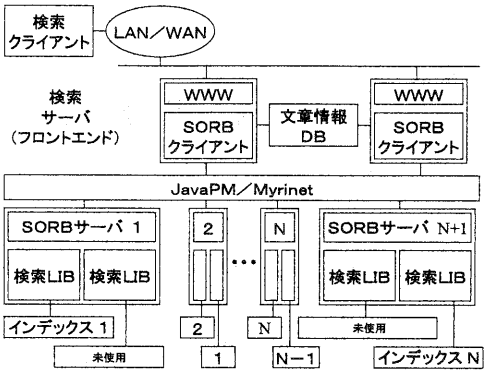

図 1．SORB によるサーバシステムの例

3. JavaPM の実装

JavaPM は、PM/Myrinet による高速通信を Java のマルチスレッド環境の中で利用する機能である。Java のソケットクラスはファクトリパターン[5]で設計されており、JavaPM はこの 1 実装として組込むことで、他の通信レイヤと同じインタフェースで利用できる。これにより、JavaPM はネットワークアプリケーション本体のプログラムを変更することなく、高速通信が可能である。

JavaPM は、Java 内でソケットエミュレート処理を行い、JNI[2]を経由して PM ライブラリを呼出す実装にした。PM ライブラリを利用するメソッドは、Java の持つ排他制御によりマルチスレッド対応を行う。PM はメッセージの順序制御、到達確認の機能を持っている。そこで、JavaPM のソケットエミュレート処理は、1)PM の最大転送サイズの制限を越えたメッセージを処理するためのパケットの細分化と再組立て処理、2)メッセージをポート番号によりアプリケーションへ分配する処理、3)TCP 仮想回線の接続／切断の処理、を行う。JavaPM は、ソケットエミュレート処理

のために 32 バイトのヘッダが必要であり、これは通信のオーバヘッドとなる。

3．1 送信処理概要

アプリケーションからの送信要求は、アプリケーションのスレッドで処理される。送信要求は、ソケットエミュレート処理した後、JNI を経由して PM の送信バッファにコピーして送信する。

PM の送信バッファが獲得できない場合には、通信相手側の受信処理が停滞していると考えられるので、CPU を他の処理で利用できるように、OS の最小単位である 1msec のスリープをはさんでポーリングしている。

3．2 受信処理概要

PM は受信メッセージを読出して受信バッファを開放する必要があるため、JavaPM は PM から受信メッセージを読出す専用のスレッド（受信スレッド）を生成する。読出したメッセージをアプリケーションに渡すまで蓄積しておくバッファ（蓄積バッファ）が必要であり、事前に 255 個準備している。未使用の蓄積バッファはキュー（未使用キュー）で管理し、受信メッセージを蓄積しているバッファはアプリケーション別に作成されたのキュー（ポートキュー）で管理する。

PM から読出し処理は、1)未使用キューから蓄積バッファを取得、2)JNI を経由して受信メッセージを蓄積バッファにコピー、3)ヘッダ情報に従ってソケットエミュレート処理をしてポートキューに接続の処理を繰返す。受信メッセージが到着していない場合には、読出し処理は 20 回ポーリングした後、PM でブロックされる。

アプリケーションの読出し処理は、1)ポートキューから蓄積バッファを取得、2)アプリケーションの領域にコピー、3)未使用キューに接続を行う。ポートキューが空である場合にはブロックされる。

受信処理はスレッド切替と 2 回のコピーのため、送信処理に比べオーバヘッドが大きい。そこで、アプリケーションのスレッドが受信待時に PM の読出し処理をすることで、スレッド切替を少なくした。さらに未使用キューは蓄積バッファを先頭に接続している。これにより、CPU のメモリキャッシュのヒット率を上げる効果がある。

4．SORB の設計と実装

SORB は Java のオブジェクト直列化機能を利用して、1)集団実行機能、2)マイグレーション機能、3)高信頼オブジェクト管理機能を拡張した耐故障性を高める ORB である。現段階では耐故障性の機能は実装されていない。

SORB の特徴は、アプリケーションが一度ルックアップしたスタブを JavaVM 内で一括管理するスタブ管理にある。スタブ管理により、アプリケーションから透過的にマイグレーション時の位置情報を制御できる。さらに、スタブの管理情報はキャッシュとしての役割を持ち、同一のリモートオブジェクトへのルックアップに対して高速に応答できる。 同じリモートオブジェクトに対して同じスタブを共同で使用するので、スタブはマルチスレッド対応の必要がある。スタブ管理機能自身も SORB の集団実行機能を利用しており、SORB を利用するクライアントでは SORB サーバが動作している必要がある。

SORB のリモート呼出しは、オブジェクト識別子・メソッド番号・引数オブジェクトを送信し、終了コード・戻りオブジェクトを受信する方法で行う。サーバ側では、登録時の名称、オブジェクト識別子、リモートオブジェクトとそのスタブを管理情報としてハッシュテーブルに登録する。

4．1 集団実行機能

集団実行機能は、1 回のメソッド呼出しで、複数のリモートオブジェクトのメソッドをマルチスレッドで同時に実行し、最後に全体の同期をとる方法で実装した。ルックアップ時にホスト名の代わりにホスト名の配列から生成されるグループ識別子を指定することで集団実行スタブを獲得する。スタブは単一実行と集団実行で同じクラスを使用する。集団実行時には引数オブジェクトを各ノー

ドにスキャッターするメソッドと戻りオブジェクトをリダクションするメソッドが実行される。

4．2　マイグレーション機能

マイグレーション機能として、リモートオブジェクトを他のノードに移動可能にする機能と、他のノードの同じ機能を持ったリモートオブジェクトに切替える機能を備える。

切替機能は、スタブ管理機能で各ノードのスタブの位置情報を変更することで実装する。

移動機能は次の手順で実装する。

1) 移動先での仮登録
   仮登録は、サーバ管理情報のリモートオブジェクトの項目を null で登録する。この状態では、オブジェクトへのルックアップは可能であるが、リモート呼出しの実行はオブジェクトが転送、初期化されるまでブロックされる。
2) スタブの位置情報修正
   移動元サーバが各クライアントのスタブの位置情報をスタブ管理機能で移動先に修正し、移動元での登録を削除する。新しいリモート呼出しはすべて移動先に対して行われる。
3) 移動元からリモートオブジェクトの転送
   移動元で実行しているリモート呼出しがすべて終了した段階で、オブジェクトを移動先に転送する。
4) 移動先でのサービス開始
   移動先でリモートオブジェクトを登録し、初期化メソッドを呼出して移動できないフィールドを初期化する。ブロックしていたリモート呼出しを実行してサービスを再開する。

4．3　高信頼オブジェクト管理機能

内部状態を持つリモートオブジェクトは、メソッドがリモート呼出しされた後に、クラスタ内にコピーを保存する。保存する方法は他のサーバにコピーをとるか、コピーを細分化してクラスタ全体に断片を分散して保存する。障害発生時には、このコピーを復元して他のサーバに登録した後、マイグレーションの切替機能を実行する。

5．　性能評価

前章までに説明した JavaPM、SORB の性能を 8 台構成の PC クラスタで測定した。各ノードの構成を表 1 に示す。PM はバージョン 1.4, JDK はバージョン 1.2.1 を使用している。

表 1. PC クラスタのノードの構成

CPU	Pentium II 400MHz
L2 Cache	512KB
Chipset	440BX
Memory	SDRAM 128MB
Copy Speed	140MB/sec
Myrinet Link Speed	160MB/sec
DMA Speed(8KB)	read 128MB/sec
	write 118MB/sec
NIC	3Com Fast EtherLink XL
OS	Windows NT4.0 Workstation SP3

5．1　JavaPM の基本性能

レイテンシ、スループットの性能測定には、同一の送受信バッファへ転送する方法をとった。これは、Java ではオブジェクトストリームを含めてバッファリングされたストリームを使用する場合に相当する。図 2、3 で、「PM」は C 言語から直接 PM ライブラリ呼ぶ場合，「JNI+PM」は Java から JNI を経由して PM ライブラリを呼んだ場合を意味している。「Thread 切替」は、受信処理がアプリケーションのスレッドで実行された場合に「なし」、受信スレッドで実行された場合には Server/Client のどちらで行われたかを表示している。

レイテンシは 2 ノード間で 4bytes メッセージのピンポンで測定した。ラウンドトリップ時間の計測結果を図 2 に示す。JavaPM のレイテンシは、38usec であった。これは、PM のレイテンシ 14.8usec、受信処理／送信処理オーバヘッドの 2 倍の合計値と考えられる。受信処理にスレッド切替が発生すると、レイテンシはスレッド切替のオーバヘッド約 25usec ずつ増加する。

スループットは 2 ノード間のバースト転送で測定した。スループットの測定結果を図 3 に示す。図中で 3 章の実装を V1.0 と示す。小さいメッセ

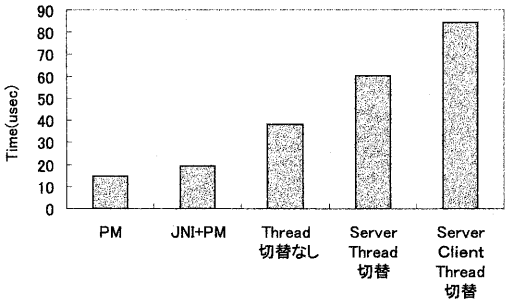

図 2. JavaPM のレイテンシ性能

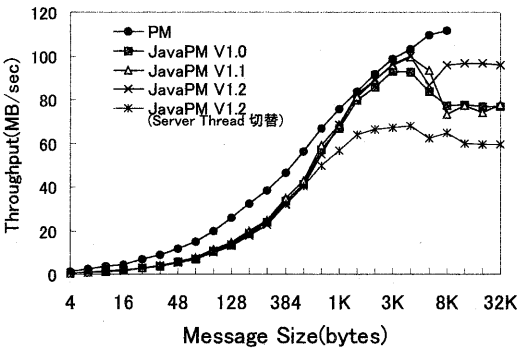

図 3. JavaPM のスループット性能

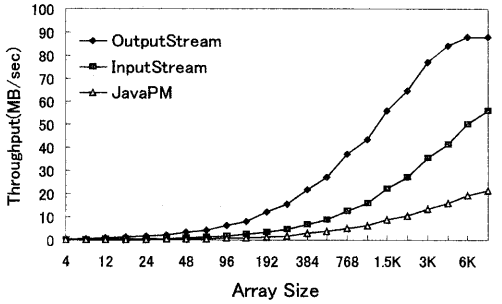

図 4. オブジェクト直列化性能

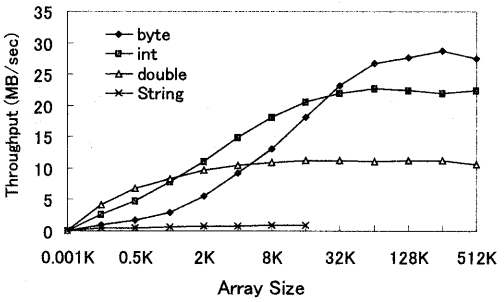

図 5. SORB のオブジェクト転送性能

注) 図 3,4 の横軸目盛は 4,8,16,32,24,32,64,48,64,96, 128,192,256,384,512,768,1K,1.5K,2K,3K,4K,6K, 8K,16K,24K,32K である.

ージをバースト転送する場合には、スループットは受信処理の性能に支配される。メッセージ長が大きい場合には、DMA のオーバヘッドが無視できなくなる。メモリコピーより DMA の速度が遅いため、次の送信を行う時にメッセージは、送信側に残って蓄積され、60KB の送信バッファを使い切ることでスリープが発生する。蓄積されたメッセージすべてを受信側に転送、受信処理する時間がスリープ時間より短いため、送受信データなしの状態が発生して、性能が劣化する。このために、図中 V1.1 では送信側で次のメッセージ送信を DMA 転送時間分待つことで改善した。

8KB 転送時の性能悪化の理由は解析中である。JavaPM ではヘッダとして 32bytes が追加され、PM のメッセージ長は 8224bytes となる。この時の PM のスループット性能は、図には表示していないが、約 64MB/sec 程度にしかならない。このため、図中 V1.2 では JavaPM の最大メッセージ長を 4KB に設定することで改善した。Java のオブジェクトストリームのバッファリングサイズは 1KB であるので SORB には影響しない。6KB 時の性能悪化はメッセージがパケット 2 個に分割されるためと考えられる。

結果として、スレッド切替が発生しない場合で最大スループット 95MB/sec、レイテンシ 38usec であり、スレッド切替が発生する場合はこのオーバヘッドが影響し、65MB/sec、60usec, 両側でスレッド切替がある場合には 85usec であった。

5．2　SORB の基本性能

メモリへの byte 配列オブジェクトの転送性能による直列化性能の測定結果を図 4 に示す。図 4 で OutputStream はメモリへのオブジェクトの直列化、InputStream はメモリからの復元化、JavaPM は JavaPM を介しての直列、復元化性能である。SORB での各種配列オブジェクトの転送性能を図 5 に示す。なお、String は Array Size

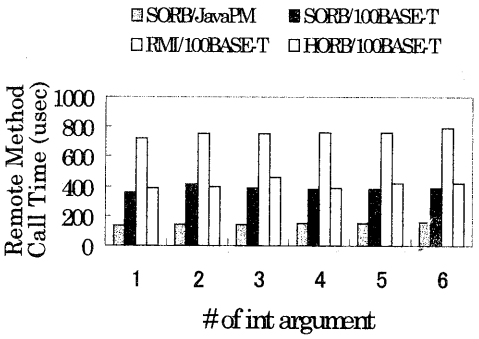

図 6. Remote Method Call の性能

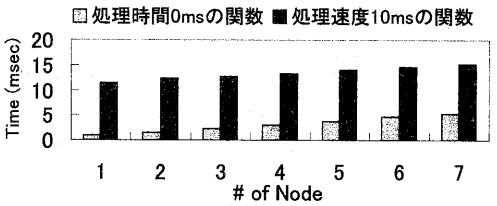

図 7. 集団実行の性能

が示す文字列を持つStringオブジェクトである。最も高速である byte 配列オブジェクトの直列化性能に対して、JavaPM のオブジェクト転送性能は追従しない。この原因は、復元化のオーバヘッドと JavaPM の受信処理時間が加算された結果が転送性能となるからである。大きな配列に対して 1KB 単位に転送される直列化データを復元するのに平均 15.4usec、JavaPM の受信処理 15.8usec の合計値 31.2usec が処理に必要となる。性能は 32.0 MB/sec 程度と計算され、測定結果とほぼ一致する。JavaPM の処理時間によりオブジェクト転送性能は悪化するが、他のオブジェクトに対しては直列化性能が落ちるため、JavaPM の影響は小さくなる。

図 6 に int を引数持つメソッドへのリモート呼出しの実行時間を示す。比較のため 100BASE-T で計測した SORB、RMI[2]、HORB[6]の測定結果を示す。HORB は HORB2.0 の SerializingIOCI の性能測定による。実行時間では SORB も HORB と同程度の性能を示し、RMI の半分程度の時間であった。SORB /100BASE-T と SORB/JavaPM のリモート呼出しの性能差は主に通信レイテンシの性能差(100BASE-T:255usec、JavaPM:38usec)によるものであると考える。

集団実行性能は 1 から 7 台ノードで int を引数持つメソッドへのリモート呼出しの実行時間で測定した。測定結果を図 7 に示す。集団実行性能は、並列度当たりのオーバヘッドが約 0.7msec あり性能が出ていない。この原因は調査中である。

6. まとめと今後の課題

我々は、高性能な PC クラスタを経済的な拡張性／耐故障性のあるサーバシステムとして利用するため、PM を Java から利用する通信レイヤ JavaPM と集団実行と耐故障性を高める機能を実装する SORB を開発して性能評価を行った。

JavaPM のスループットは、小さいメッセージ長では受信処理のオーバヘッドにより、長いメッセージ長ではスレッド切替のオーバヘッドにより PM の 1/2 程度となった。レイテンシは、スレッド切替時のオーバヘッドが大きく影響して 85usec であった。SORB のオブジェクト転送性能は最も高速な直列化である byte 配列で 30MB/sec の性能を得た。SORB のリモート呼出しの性能は、HORB と同程度、RMI の 1/2 程度の時間であった。集団実行機能の性能改善が課題である。

今後、耐故障性を実装するために、JavaPM、SORB に障害検出とそのリカバリーの機能追加をしていく予定である。

参考文献

[1] RMI: http://java.sun.com/products/jdk/rmi/

[2] Java, Technology, http://java.sun.com/

[3] http://www.myri.com/myrinet

[4] PM: High-Performance Communication Library, http://pdswww.rwcp.or.jp/pm/

[5] Erich Gamma ほか, オブジェクト指向における再利用のためのデザインパターン, ソフトバンク, 1995.

[6] HORB, http://ring.etl.go.jp/openlab/horb-j/