ハイパフォーマンス
コンピューティング 87－20
（２００１．７．２６）

Beowulfクラスタにおける高精度実行時間測定の検討と評価

早川 潔† 関口 智嗣§ 岩根 雅彦†

Beowulfクラスタシステムは,比較的安価でかつ容易に構築できるため,非常に注目を集めており,Beowulfクラスタでの通信性能をはじめとする種々の処理性能向上に関する研究が盛んに行われている.また,処理性能の高精度な測定は,より正確な評価を可能にし,通信時間の隠蔽などといった並列処理の効率化を可能にする.一般的に処理性能を測定するためには,各ノードの実行開始時刻を正確に揃えなければならない.一般的なBeowulfクラスタでは,各ノードの実行開始時刻を揃えるために,MPIなどの通信ライブラリのBarrier関数が用いられる.しかし,Beowulfクラスタに実装されるBarrierでは,ある程度の誤差が生じてしまう.そこで,本稿では,PCをベースしたBeowulfクラスタシステム（SCCB-Cluster system）における高精度実行時間測定システムの検討を行った.高精度な測定を可能にするために,Beowulfクラスタに高速なバリア同期を可能にするSCCボードを搭載した.また,そのSCCボードの中にクロックカウンタを搭載し,疑似的なグローバルクロックを実装する.性能評価として,Beowulfクラスタのcollective通信性能を測定した.SCCボードでの高速なバリア同期を用いた実行時間測定値は,Ethernetを使用したMPI_Barrierを用いた測定値より,安定し,かつ,短い値を示したものあった.

Accurate Performance Measurement System for Beowulf Cluster Systems

Kiyoshi Hayakawa †, Satoshi Sekiguchi §, Masahiko Iwane †

Beowulf cluster consisted of commodity parts, such as PCs and 100base/TX LAN card, is the most remarkable parallel computer system. Collective communications using MPI are the most integral packet forwarding methods on the cluster computing. Accurate performance analysis of collective communication is useful on performance evaluation and prediction of Beowulf cluster system. In order to measure execution time accurately, each node have to take the first step with execution by barrier. But it is difficult for each node to take the first step with execution each other, since it receives the packet indicating barrier completion through Ethernet (i.e. MPI_Barrier) in different time.
This paper describes the Beowulf cluster system (SCCB cluster) that allows us to measure execution time accurately. SCC (Synchronization Communication Controller) was implemented in this cluster system. SCC is able to finish executing barrier less than 10us with 32 nodes. As the performance analysis of MPI collective communication using barrier that SCC performs (SCC_Barrier), performance of MPI collective communication of SCC_Barrier is less fluctuation than that of MPI_Barrier.

1 はじめに

PCや100base/TX Ethernetなどの汎用部品で構成されたBeowulfクラスタシステムは,比較的安価でかつ容易に並列処理システムを構築できるため,注目を集めている.また,市販マイクロプロセッサの性能が急激に向上しているので,そのプロセッサを使用するBeowulfクラスタシステムはより高速な並列処理を可能にしている.よって,Beowulfクラスタでの通信性能をはじめとする種々の処理性能に関する研究[2][3]は重要な研究の一つであり,処理性能の高精度な測定は,より正確な評価を可能にすることに役立つ[6][5].また,処理時間を正確に測定し,その測定値を基にしてアプリケーション内の各処理の実行時間を予測することにより,通信時間の隠蔽などといった並列処理の効率化を可能にする.

一般的に処理性能を測定するためには,各ノードの実行開始時刻を正確に揃えなければならな

†:九州工業大学 工学部 電気工学科
Department of Electronic and Computer engineering Kyushu Institute of Technology
§: 産業技術総合研究所
National Institute of Advanced Industrial Science and Technology

い. 一般的に, 各ノードの実行開始時刻を揃えるために,MPI などの通信ライブラリの Barrier 関数 (MPI_Barrier) が用いられる.MPI などの通信ライブラリは,Ethernet を使用して Barrier を行っている. しかし,Ethernet を使用した Barrier 同期では, パケット転送時間や TCP/IP などの各種通信レイヤでの処理時間のオーバーヘッドが大きいため, ある程度の精度（数 100μs 程度）では揃えられるが, それ以上の精度は望めない.

そこで, 本稿では,PC をベースした 32 台の Beowulf クラスタシステム（SCCB-Cluster system）における高精度実行時間測定システムの検討を行った. 高精度な測定を可能にするために,Beowulf クラスタに高速なバリア同期を可能にする SCC ボードを搭載した. また, その SCC ボードの中にクロックカウンタを搭載し, 疑似的なグローバルクロックを実装する.

性能評価として,SCCB-Cluster の collective 通信性能を測定した.SCC ボードでの高速なバリア同期を用いた実行時間測定値と Ethernet での MPI_Barrier 同期を用いた実行時間測定値とを比較した.

2 SCCB-Cluster システム

図 1に SCCB[1]-Cluster のシステム構成を示す.SCCB クラスタは PC ボード（PICMG 規格のボード CPU : PentiumIII600MHz）32 台を 100Base/TX の Switching HUB で結合したシステムである. 各ノードには,SCC[2][1] と呼ばれる高速バリアを可能にするボードが実装されている. 各 SCC は Sync-Comm Network と呼ばれる同期・通信用ネットワークで結合されている.OS は Linux(kernel version:2.2.15), 通信ライブラリは MPICH(version 1.2.0) である.

3 SCC ボード

SCC ボードは, メッセージパッシング型の通信処理とバリアを拡張した同期処理を高速に行うための PCI ボードである [1].SCC は, 高スループットを要求する処理は従来のネットワークボードにまかせ, 低レイテンシが要求される同期や小量データパケットのみを処理することにより良い並列処理が実現する. 本稿では,SCC の同期処理機能のみを使用して, 高精度実行時間測定システムを検討する.

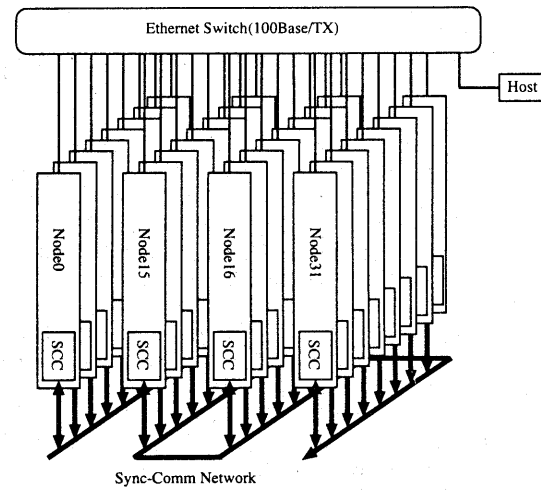

図 1: SCCB-Cluster

3.1 SCC ボードの構成

SCC ボードは,Control LSI および sync-comm newtwork 用コネクタ（Link_A および Link_B）のみで構成されている.Control LSI は, 同期・通信処理をハードウェアで実装することにより高速化を実現する.sync-comm newtwork 用コネクタは,sync-comm newtwork 用コネクタの Link_A と Link_B をデージーチェーンで接続する（図2参照）ことにより, ハブ等の外部接続装置を必要とせずに,sync-comm newtwork が構成可能である.

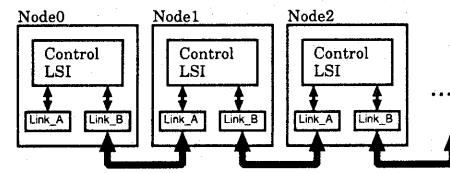

図 2: Sync-comm Network

3.2 SCC の Control LSI

図 3に Control LSI 内のブロック図を示す.Control LSI は,PCI バス-内部バスブリッジ（PCI

[1]Sync-Comm Controller Beowulf
[2]Synchronization Communication Controller

BUS-Internal BUS Bridge），同期制御部（Sync Control），通信制御部（Comm Control），疑似グローバルクロック部 (Pseudo-global clock), およびネットワーク制御部（Network Control）で構成される.

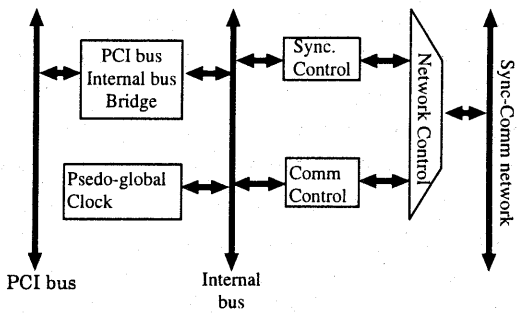

図 3: Control LSI 内のブロック図

PCIバス・内部バスブリッジは,PCIバスプロトコルと内部バスプロトコルをインターフェースし,PCIのマスター／ターゲット機能もサポートする. 同期制御部は,sync-comm network を使用した同期処理アクセスを行い, 任意参加バリア,Fuzzy バリアおよび RBC 同期機構 [1] の同期を処理する. 通信制御部は, パケットデータ処理を行う. 疑似グローバルクロックは, バリア制御部と連携し, 各ノードの実行開始時刻を揃えると同時にクロックカウンタをスタートさせるなどの高精度実行時間測定に必要な処理をハードウェアで行う. ネットワーク制御部は, 同期信号と通信信号とを sync-comm network 上で融合させるための制御を行う.

3.3 sync-comm network での同期信号

同期時における sync-comm network の信号線は, 図 4のように接続される. 同期時の信号線は, 最大 16 台で構成されるマイクロクラスタ内での同期信号線とそのマイクロクラスタ間で同期を行うための同期信号線に分かれる.

同期成立の検出は, まず, マイクロクラスタ内で同期成立を検出し, その後にマイクロクラスタ間で同期成立を検出する. マイクロクラスタ内では, sync_n (n はマイクロクラスタ内でのノード番号) を使用し, マイクロクラスタを構成する各ノード間の同期成立を検出する.

sync_n 信号は, マイクロクラスタ内ノード番号 n のノードが同期ポイントに到達していることを示す信号である. この信号線は, マイクロクラスタ内のみのバス接続である.

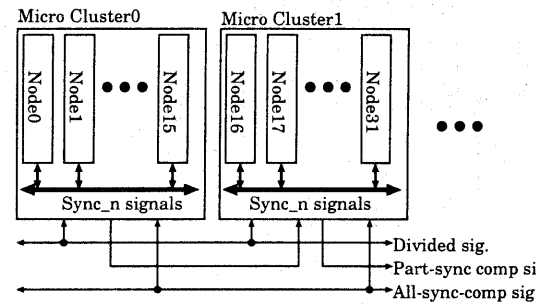

図 4: Sync-Comm Network の同期信号

マイクロクラスタ間では,Divided 信号,Part-sync-comp 信号,All-sync-comp 信号を使用して, クラスタ間の同期成立を検出する. Divided 信号はターゲットクラスタシステムがマイクロクラスタに分割されていること（ノード数が 17 台以上であること）を示す信号,Part-sync-comp 信号はクラスタ内の同期が成立したことを示す信号,All-sync-comp 信号は, クラスタ全体の同期が成立したことを示す信号である.Divided 信号線,All-sync-comp 信号線は, クラスタ全体のバス接続である. Part-sync-comp 信号線は, デージチェーン接続である.

3.4 同期制御部の機能

同期制御部は, 同期成立検出回路 (synchronization completion detector) および R-BC 回路で構成される（図 5参照）.

同期成立検出回路には, マクロクラスタ内での任意参加バリアをサポートするために, 同期グループレジスタ（sync-group reg）（16 ビット）が用意されている. また, 同期の成立を知らせるために, 同期フラグ（S-flag）が用意されている.

バリア同期の動作を以下に示す.

1. バリア同期ポイントに到達したノードは,S-flag をセットする.

2. sync. completion detector がバリア同期が成立したかどうかチェックする.

3. 同期が成立したら,S-flag をリセットする.

4. バリアポイントで実行を中断しているノードは,S-flag のリセット後, 実行を再開する.

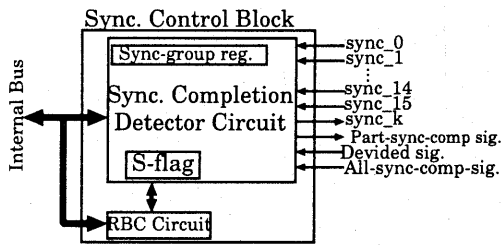

図 5: 同期制御部の構成

4 疑似グローバルクロックカウンタシステム

一般的に, 実行時間測定には, 各ノードのマザーボード内に搭載されているクロックカウンタを使用する. それらのカウンタを使用した実行時間測定には, ある程度誤差が生じてしまう. あるノード単独で処理する処理時間を測定するには, 誤差は生じないが, 通信処理などの他のノードと協調して処理する処理時間を正確に測定するには, グローバルクロックカウンタシステムを使用して実行時間を測定する必要がある.

グローバルクロックシステムを実装してないシステムの場合, 各ノードの処理開始時刻を揃えた後, 各ノードのクロックカウンタをスタートさせ実行時間を測定する. 一般的に, 各ノードの実行開始時刻を揃えるために,MPI などの通信ライブラリの Barrier 関数 (MPI_Barrier) が用いられる.MPI などの通信ライブラリは, Ethernet を使用して Barrier を行っている. しかし,Ethernet を使用した Barrier 同期では, パケット転送時間や TCP/IP などの各種通信レイヤでの処理時間のオーバーヘッドが大きく, ある程度の誤差で揃えた後に, クロックカウンタをスタートさせざるおえない. よって, 各ノードのクロックカウンタによる実行時間測定には, ある程度の誤差が生じてしまう. 実行時間測定の誤差を低減させるため,SCC に疑似グローバルシステムを実装することを検討する.

4.1 疑似グローバルクロックカウンタシステムの構成

図 6に疑似グローバルクロックカウンタシステムの構成を示す. 疑似グローバルシステムは,25MHz のクロック (OSC) , クロックカウンタ (32bit), タイムスタンプメモリおよびカウンタコントローラで構成されている. カウンタコントローラがバリア同期とカウントスタートを連動して行う. つまり, カウンタコントローラが Synchronization Control 内の S-flag を常に監視し, S-flag がリセットさせたら, カウンタをスタートさせる. また, ユーザープログラムの要求にしたがって, カウンタの値をタイムスタンプメモリに格納したり, 読み出したりする. シミュレーション結果から, このシステムを使用することにより, 誤差が 240ns まで低減する。

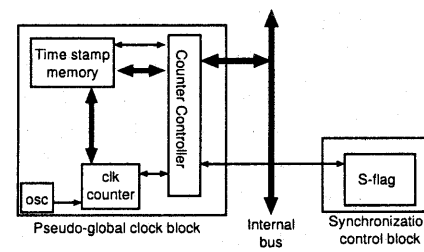

図 6: 疑似グローバルクロックシステム

5 SCC 同期ライブラリ

SCC を使用したバリア同期を行うために,SCC 用の同期ライブラリを作成した (図 7参照) .SCC ライブラリには, SCC_Init(), SCC_finalize(), SCC_Barrier(), SCC_FuzzyBarrier() などが実装されている. SCC ライブラリは,SCC のデバイスドライバに対してシステムコールを行うことにより,SCC にアクセスする. デバイスドライバは,PCI バスを介して,SCC から S-flag などの情報を読み出し, バリアグループなどの情報を書き込む.

SCC ライブラリは MPICH プログラムの中で使用することができる. つまり,MPI のプログラムの中に,SCC ライブラリの各関数を挿入することができる.Ethernet を使用したバリア同期を行いたい場

合,MPI_Barrier を使用し,SCC を使用した高速バリア同期を行いたい場合,SCC_Barrier を使用する.

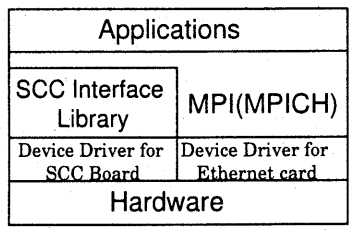

図 7: SCC ライブラリのレイヤ構成

6 collective 通信の性能測定

高精度測定の評価として,SCC での高速バリア同期を使用した collective 通信の測定を行った. 今回の測定は, 疑似グローバルクロックシステムを使用するかわりに,SCC での高速バリアおよびマザーボードのクロックカウンタを使用した性能測定を行った. つまり, 高速バリア同期を行った後にマザーボードのカウンタをスタートさせるという動作をソフトウェアで行った.

6.1 高速バリア同期の性能評価

図 8にノード数と SCC での高速バリア同期レイテンシの関係を示す.SCC を使用した高速バリア同期 (SCC_Barrier) の性能と Ethernet を使用したバリア同期 (MPI_Barrier) の性能を比較した.SCC を使用したバリア同期のレイテンシは,32 ノードまではノード数に関係なく,1.6μs または,3.3 μs である. このシステムは, ほぼ 1.5μs に一回の割合で同期フラグにアクセスできる. しかし, ハードウェア見積りに比べて非常に遅いので, 図 8に示す結果を得たと推測される. バリア同期のハードウェアレイテンシの見積りが 64 ノードでも 3μs を越えないため,32 ノード以上のバリア同期でも,3.3μs のレイテンシで処理できると推測される.

一方,MPI_Barrier は, ノード数が増えるにつれて増加し,32 ノードでは, 本同期コントローラのバリア同期レイテンシに比べほぼ 200 倍のレイテンシを得た.

6.2 MPI collective 通信における高精度測定の評価

SCC を使用した高速バリア同期で, 開始時刻を揃えて測定した MPIcollective 通信の測定結果を図 9, 図 10, 図 11, および図 12に示す. 各図では,Ethernet を使用した MPI_Barrier で, 開始時刻を揃えて測定した MPIcollective 通信の測定結果と比較した.

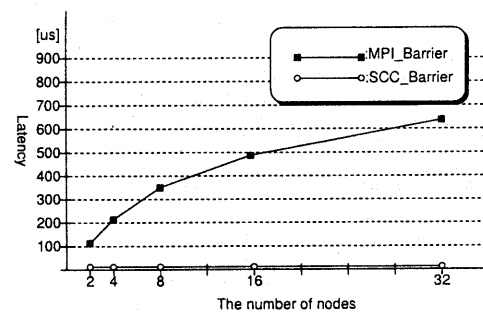

図 8: SCC を使用したバリア同期のレイテンシ

高速バリアで開始時刻を揃えて測定した測定結果は, 全てほぼ log(n) カーブを示している. 一方,Ethernet で開始時刻を揃えた測定結果は,log(n) カーブを示していないものもあり, プロセッサ台数が増えているにも関わらず, 実行時間が減少してしまうという不可解な結果を示している.

MPI_Bcast と MPI_Reduce の一部の結果は以外, 高速バリアで開始時刻を揃えて測定した測定結果は,Ethernet で開始時刻を揃えた結果よりも実行時間が短い.

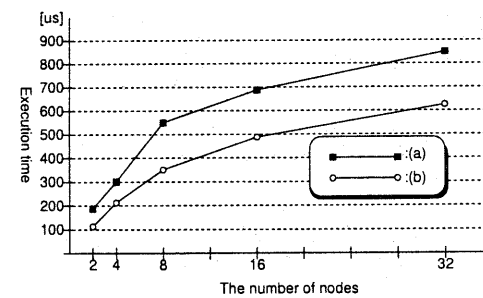

図 9: MPI_Barrier の測定結果. (a):MPI_Barrier で開始時刻を揃えて測定, (b):SCC での高速バリア同期で開始時刻を揃えて測定.

7 おわりに

PCをベースした32台のBeowulfクラスタシステム（SCCB-Cluster system）における高精度実行時間測定システムの検討を行った．高精度な測定を可能にするために,Beowulfクラスタに高速なバリア同期を可能にするSCCボードを搭載した．また,そのSCCボードの中にクロックカウンタを搭載し,疑似的なグローバルクロックを実装することを検討した．

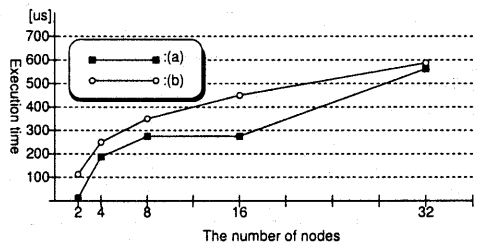

図 10: MPI_Bcastの測定結果

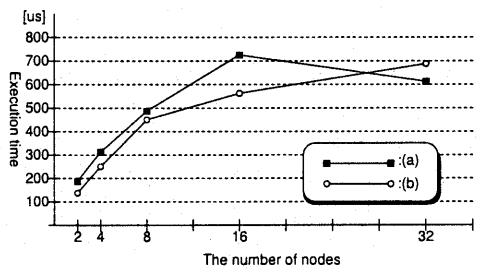

図 11: MPI_Reduceの測定結果

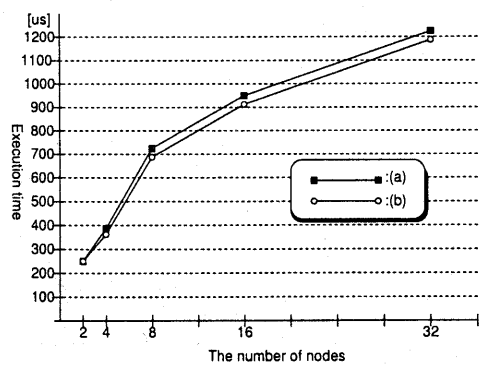

図 12: MPI_Allreduceの測定結果

性能評価として,Beowulfクラスタのcollective通信性能を測定した．高精度な性能測定に必要なSCCでの高速バリア同期のレイテンシを測定した結果,32台まで,台数に関わらず,1.6μsまたは3.3μsの高速なバリア同期を実現できた．シミュレーションでは,32台以上でも,最大3.3μsの値を保持できる結果を得ている．この高速バリア同期を用いて,collective通信の性能測定を行った結果,高速バリアを用いた性能測定では,比較的誤差が少なく測定できていると推測できる。

今後の課題は,疑似グローバルクロックを実装することと,その疑似グローバルクロックの値を使用した処理時間表示ツールを作成することである．

参考文献

[1] Kiyosh Hayakawa, Satoshi Sekiguchi“Design and Implementation of a Synchronization and Communication Controller for Cluster Computing Systems,” Proc. 4th Intel. Conf. High Performance Computing in Asia-Pacific Region, vol.I, pp76-81, May.2000.

[2] Nanette Boden, Danny Cohen, Robert Feldrman, Alan Kulawik, Chrle Seitz, Jakov Seizovic, and Wen-King Su, “Myrinet – A Gigabit per Second Local Area Network”,*IEEE Micro*, vol.15, No.1, pp.29-36, Feb.1995.

[3] H.Tezuka, A.Hori, Y.Ishikawa, and M.Sato, “ PM: An Operating System Coordinated High Performance Communication Library.”,*High-Performance Computing and Networking*, vol.1225, pp.708-717, Apr.1997.

[4] S.Pakin, M.Lauria, and A.Chein, “High Performance Messaging on Workstations: Illinois Fast Messages (FM) for Myrinet”,In Proc. of Supercomputing ’95, 1995.

[5] 田中良夫, 久保田和人, 佐藤三久, 関口智嗣 “並列アルゴリズムにおけるCollective通信の性能比較”, 情報処理学会研究報告,96-HPC-62, pp.19-26, Aug.1996.

[6] 益口摩紀, 建部修見, 佐藤三久, 関口智嗣, 長嶋雲兵 “並列システム性能の視覚的解析とその性能評価”, 情報処理学会研究報告,99-HPC-75, pp.73-78, Mar.1999.