様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

平成　３０　年　　８　月　２２　日現在

機関番号：　１４４０１
研究種目：　基盤研究(C)（一般）
研究期間：　2014～2017
課題番号：　２６３３０１４５

研究課題名（和文）OpenFlow結合網での高速MPI通信に向けたフロー制御命令列ジェネレータ

研究課題名（英文）Flow Control Sequence Generator for Fast MPI Communication on an OpenFlow Interconnect

研究代表者

伊達　進（Date, Susumu）

大阪大学・サイバーメディアセンター・准教授

研究者番号：２０３４６１７５

交付決定額（研究期間全体）：（直接経費）　　3,600,000円

研究成果の概要（和文）：本研究では、MPIプログラムの特性とSDN(Software Defined Networking)のネットワークプログラミング性を連動させ、MPIプログラムを高速に実行する並列分散計算実行環境を実現した。より具体的には、OpenFlow結合網を有する計算クラスタ上で、MPI通信によって発生するネットワークフロー系列をOpenFlowコントローラでプログラム制御することで、本研究での提案手法が、MPI通信、特にMPI_Bcast、MPI_Allreduce等の集合通信に要する時間の縮減に有効であることを示した。

研究成果の概要（英文）：This research has realized a parallel distributed computing environment that accelerates MPI execution, taking advantage of network programmability brought by SDN (Software Defined Networking) and MPI program characteristics. In more detail, the proposed method in this paper dynamically controls a sequence of network flows on OpenFlow Controller in a software-programming manner. We have shown that the proposed method in this research is effective to reduction of execution time for MPI collective communications such as MPI_Bcast and MPI_Allreduce.

研究分野：　広域分散計算

キーワード：　MPI　SDN

１．研究開始当初の背景

(1) 近年、並列分散プログラミングライブラリ MPI(Message Passing Interface)の必要性と重要性が急激に高まりつつある。MPI は、1 対 1 通信、複数プロセスが関与する集合通信に関する API (Application Programming Interface) を提供し、開発者が高性能計算を行う計算機システムのネットワーク構成や特性を意識することなく、複数のプロセス間でメッセージを複雑に交換する分散並列プログラムを比較的容易に開発可能とする。このようなプログラミングの容易性と、近年の高性能計算機システムのクラスタシステムへのアーキテクチャ変化によって、複数のノード上に分散するプロセッサコアを利用するための分散メモリプログラミングを可能にする MPI は、今日の計算機システム上で高性能計算を行う上でますます重要かつ必要不可欠な役割を担っている。

(2) 大規模クラスタシステム上で MPI プログラムを高速に実行できるかどうかは、プログラム内で実行される MPI 通信、特に、複数のプロセスが関与する集合通信に要する時間をどれほど短縮できるかに大きく依存する。今日までの先行研究は、クラスタシステムのインターコネクトは MPI プログラムから制御できない静的なネットワーク資源である、という前提に基づいている。

(3) 従来のネットワーキング機能をパケット転送機能とその制御機能に分離した、新しいネットワークアーキテクチャのコンセプト SDN (Software Defined Networking) へのネットワーク研究者、技術者らの関心と期待が急速に高まっている。

２．研究の目的

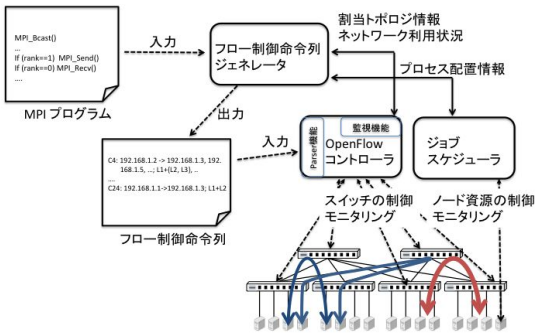


図１：高速 MPI 通信にむけたフロー制御命令列ジェネレータ.

本研究では、クラスタシステムのインターコネクトを動的に制御可能なネットワーク資源と捉え、MPI プログラム特性と SDN のネットワークプログラミング性をシームレスに連携させるという新しいアプローチで、MPI 通信の高速化を実現することを目指す。具体的には、本研究では、インターコネクトとして OpenFlow 網を想定し、MPI プログラム実行時に発生する MPI 通信系列およびそれらの特性、インターコネクトトポロジ、利用状況、MPI プロセス配置情報を基に、MPI 通信によって発生するネットワークフロー系列を OpenFlow コントローラでプログラム制御することにより、MPI 通信ひいては MPI プログラムの高速実行を可能とする並列分散計算実行環境(図１)を実現する。特に、本申請では、OpenFlow コントローラおよびジョブスケジューラと連携し、入力となる MPI プログラムより MPI 通信系列および特性を抽出し、それらより OpenFlow コントローラが入力として利用するフロー制御命令列を生成する、OpenFlow 網での高速 MPI 通信を実現するためのフロー制御命令列ジェネレータを中核技術として開発する。

３．研究の方法

本申請研究では、上述の目的達成のために、マイルストーンとなる以下の 3 課題を設定し、研究を推進する。

[課題 1] MPI 通信系列・特性に基づくフロー制御アルゴリズムの設計・実装

本課題では、MPI 実行時、どの MPI プロセスがどのプロセスとどのように通信するか？を抽出し、インターコネクトの利用状況を加味しつつ、利用する経路等を制御するフロー制御アルゴリズムを設計・実装する。特に、集合通信 MPI_Alltoall、MPI_Allreduce、MPI_Bcast を中心に着目する。

[課題 2] フロー制御命令列の記述方法、フロー制御命令列生成機能の設計・実装

本課題では、OpenFlow コントローラがインターコネクトを形成するスイッチ群の制御を行う際の入力となるフロー制御命令列の記述方法を設計するとともに、[課題 1]で設計されるフロー制御アルゴリズムが MPI プログラム実行時のインターコネクト利用状況、プロセス配置情報を基に算出するフロー制御手順に基づきフロー制御命令列を生成する機能を設計・実装する。

[課題 3]　実環境での性能評価と有用性検証

本課題では、図１に示す並列分散環境を実際に構築し、当該環境上で MPI_Alltoall、MPI_Allreduce、MPI_Bcast を中心に、集合通信等の通信性能を評価する。さらに、Intel MPI Benchmark や、実際の科学計算で利用されている MPI プログラムを利用し、本研究成果の有用性・実用性を検証する。

４．研究成果

本研究では、上述のマイルストーンとなる 3 課題をステップバイステップに達成することにより、2 で上述した目的を実現した。以下、本研究で実現したフロー制御命令列ジェネレータについて記す。

上述したように、本研究では、MPI プログラム内の MPI 集合通信の通信パターンに、SDN によって実現されるネットワークプログラ

ミング性を適用し、MPI 集合通信の高速化、ひいては、MPI 計算の高速化を担う。本目的のために、OpenFlow 網を相互結合網として有するクラスタシステム内において MPI 通信に起因して発生するパケットフローを OpenFlow コントローラ上で制御するためのフロー制御命令列を生成するフロー制御命令列を設計・開発した。

課題：

制御命令列は、MPI アプリケーション全体のためのパケットフローを OpenFlow コントローラが制御するための命令列である。これを実現するために、OpenFlow コントローラはどの MPI 集合通信がいつどのプロセス上で実行されるのかといった MPI プログラムの包括的な情報を必要とする。そのため、制御命令列は、少なくとも次に示す情報が必要となる。

1. MPI 集合通信の時系列リスト
2. MPI 集合通信のスコープ情報
3. グループとプロセウ間の関係に関する情報
4. MPI プログラムで利用されるプロセスの IP アドレス
5. データを送信するプロセスの rank 情報

制御命令列ジェネレータを生成するためには、上述の情報をどのように抽出し、抽出した情報から OpenFlow コントローラへの入力としての制御命令列をどのように生成するかが技術課題となる。

提案手法の設計と実装；

本研究では、MPI 集合通信の特性を抽出するために、実行時のログを利用する手法を選択した。しかし、上述したような必要不可欠な情報はログに記録される一方、MPI プログラム内で実行される MPI 集合通信のスコープ情報や順序といった情報は不明のままである。このため、本研究では、ログファイルを解析することによって不明な情報を収集するプログラム Log Analyzer を実装した。

以下、実装内容について記す。MPI の拡張である MPE(MPI Parallel Environment は、MPI 開発者のために有用な数多くの機能を提供している。それらの機能には、MPI プログラムの情報を可視化、リアルタムアニメーション向けのログファイルとして記録するプロファイリングライブラリが含まれている。本研究では、CLOG-2 型式ログより CLOG_2PRINT プログラムにより ASCII テキスト型式に変換する。

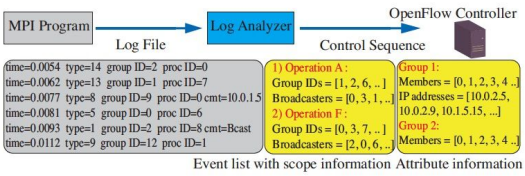


図 2: フロー制御命令列の動作.

本研究では、制御命令列生成のために必要となる情報を抽出するために、MPI プログラムのログ、具体的には、CLOG-2 形式でのログを採用した。ログを通じて取得できない MPI プロセスの IP アドレス、データ送信を行うプロセスの rank 情報の取得には、MPI プログラム内にカスタマイズイベントを定義する方法でログファイルに記録するようにした。

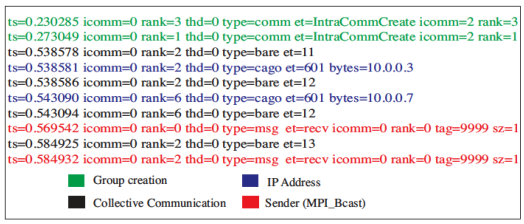


図 3: フロー制御命令列生成に利用されるログの例。

```
int root = 0;
// Initializing MPE logging
MPE_Init_log();
// Custom events
event1 = MPE_Log_get_event_number ();
event2 = MPE_Log_get_event_number ();
// Recording a custom event with IP address
MPE_Log_event (event2, 1, my_ip_addr);
MPI_Bcast (&lin, 4, MPI_INT, root, MPI_COMM_WORLD);
// Identifying the process that broadcasts
MPE_Log_receive (root, 9999, 1);
// Finalize MPE logging
MPE_Finish_log ("log_test01");
```

図 4: カスタムイベントをログするためのコード例.

図 3 にユーザ定義ロギングの使用を通じて生成されたログファイルの一部を示す。また、図 4 にユーザ定義ロギングの擬似コードを示す。MPI プロセスが実行される計算ノードの IP アドレスを記録するために、各プロセスはコメントとして IP アドレスをカスタムイベントを利用する。データを送信するプロセスの rank を特定するために、あるプロセスからのメッセージ受信のログを取得する関数を call する。図 4 中の最初の行は MPE を初期化するために利用される。その後の 2 行はユーザが定義するカスタムイベントを宣言している。MPE_Log_event は global rank および local rank をプロセスの IP アドレスとともに記録させる。その後、MPI_Bcast が呼びされると、MPI_Log_receive が呼び出されデータを送信したプロセスを特定する。MPE_Log_receive の入力引数 9999 はブロードキャストメッセージを区別するために使われるメッセージタグである。最終行は MPE の終了処理を行う。

上述のようにして取得されたログデータから、Log Analyzer は以下の構造をもつ制御命令列を生成する

1. イベントリスト
   それぞれのリストは以下の構成要素か

らなる。
- 型(文字列)
- グループリスト(整数)
- ソース(整数)（オプション）

2. グループ辞書
3. アドレス辞書

図 4 に MPI 実行期間中に発生した MPI 集合通信イベントのリストを示す。最上部のセグメントは、イベントリストを表し、制御命令列の種要素である。それぞれのイベントリストは、集合通信の種別を表す type、集合通信の有効範囲である scope 情報、データを分散するプロセスの rank から構成される。第 2 セグメントのグループリスト、および、第 3 のアドレス辞書は、MPI プログラムの通信グループの global rank およびアドレス情報を保管する。

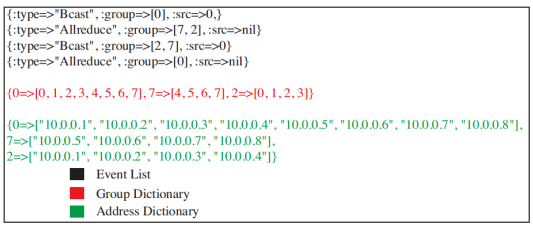


図 4: 制御命令列の一例.

評価：

本研究では、制御命令列にかかる時間の計測実験と検証試験により、提案手法を評価した。

(1)制御命令列生成時間の計測

MPI 実行を行う 24 台のスレーブノード、および、制御命令列生成を行う 1 台のマスターノードからなる実クラスタシステムを用いて評価を行った。実験に用いた計算機は表 1 および 2 の通りである。

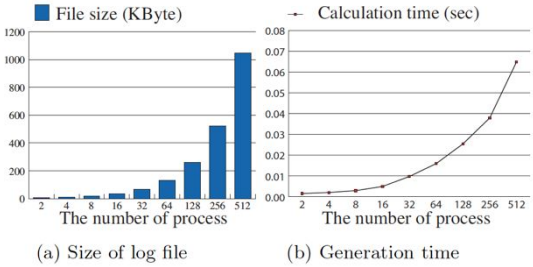
表 1: マスターノードハードウェア仕様.

Model	SGI® Rackable™ Half-Depth Server C2005V
CPU	Intel® Xeon® E5-2630L 2.00GHz 6core x2
Memory	64GB (DDR3 1600 8GB DIMM x8)
NW I/F	Gigabit Ethernet port x8
HDD	1TB SATA 7200rpm x8 (HW RAID6)

表 2: スレーブノードハードウェア仕様.

Model	SGI® Rackable™ Half-Depth Server C1001
CPU	Intel® Xeon® E5-2620L 2.00GHz 6core x2
Memory	64GB (DDR3 1600 8GB DIMM x8)
NW I/F	Gigabit Ethernet port x7
HDD	500GB SATA 7200rpm x1

また、計測実感には、2 つの global routine（MPI_Bcast, MPI_Allreduce）および 2 つの local routine(MPI_Bcast, MPI_Allreduce) からなるプログラムを利用した。実験では、15 回フロー制御命令列生成を実行し、ログファイルサイズ、ログ生成時間を生成した。



(a) Size of log file (b) Generation time

図 5: 制御命令列の評価.

図 5 に評価結果を示す。図より、制御命令列生成に要する時間は、ログファイルのサイズはログファイルサイズに比例することがわかる。したがって、冗長な情報の削減によってフォルファイルを削減することにより、制御命令列生成に要する時間の改善が見込まれる。

(2) 検証実験

SDN-enhanced MPI 集合通信が、制御命令列ジェネレータによって生成された命令列によって正しく動作するかを検証した。なお、現段階では、生成された制御命令列とネットワーク制御の同期の問題は解決していないため、MPI プログラム中より同期をとる方法を利用している。その結果、正確な命令列が制御命令列ジェネレータによって生成され、また、その生成された制御命令列を利用して行われた計算結果も正確であることを確認した。

まとめと今後の課題：

本研究では、フロー制御命令列を生成するフロー制御命令列ジェネレータを提案・設計・実装し、その有効性を評価した。今後の課題としては、生成された制御命令列とネットワーク制御の同期の問題があげられる。すなわち、計算と通信の連携を行う手法の研究開発が今後の取り組むべき課題としてあげられる。

5. 主な発表論文等

（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計 4 件）

(1) Keichi Takahashi, Susumu Date, Dashdavaa Khureltulga, Yoshiyuki Kido, Hiroaki Yamanaka, Eiji Kawai, Shinji Shimojo, “UnisonFlow: A Software-Defined Coordination Mechanism for Message-Passing Communication and

Computation", IEEE Access, vol. 6, no. 1, pp. 23372-23382, Apr. 2018. [10.1109/ACCESS.2018.2829532]

(2) Susumu Date, Takashi Yoshikawa, Kazunori Nozaki, Yasuhiro Watashiba, Yoshiyuki Kido, Masahiko Takahashi, Masaya Muraki, Shinji Shimojo, "Towards a Software Defined Secure Data Staging Mechanism", Sustained Simulation Performance 2017 (WSSP 2017), pp.15-24, Springer, 2017. [DOI: 10.1007/978-3-319-66896-3_2]

(3) Susumu Date, Hirotake Abe, Dashdavaa Khureltulga, Keichi Takahashi, Yoshiyuki Kido, Yasuhiro Watashiba, Pongsakorn U-chupala, Kohei Ichikawa, Hiroaki Yamanaka, Eiji Kawai, Shinji Shimojo, "SDN-accelerated HPC Infrastructure for Scientific Research", International Journal of Information Technology, Volume 22, Number 01, 2016.

(4) Susumu Date, Yoshiyuki Kido, Khureltulga Dashdavaa, Keichi Takahashi, Yasuhiro Watashiba, Shinji Shimojo, "Toward Flexible Supercomputing and Visualization System", M. M. Resch et al. (eds.), Sustained Simulation Performance 2015, Springer, 2015. [DOI:10.1007/978-3-319-20340-9_7]

〔学会発表〕（計 13 件）

(1) Keichi Takahashi, Susumu Date, Khureltulga Dashdavaa, Yoshiyuki Kido, Shinji Shimojo, "PFAnalyzer: A Toolset for Analyzing Application-aware Dynamic Interconnects", the Monitoring and Analysis for High Performance Computing Systems Plus Applications (HPCMASPA) Workshop, Cluster 2017, pp. 789-796, Honolulu, Hawaii, Sep. 2017. [DOI 10.1109/CLUSTER.2017.18]

(2) Arata Endo, Ryoichi Jingai, Susumu Date, Yoshiyuki Kido, Shinji Shimojo, "Evaluation of SDN-based Conflict Avoidance between Data Staging and Inter-Process Communication", The 2017 International Conference on High Performance Computing & Simulation (HPCS 2017), pp. 267-273, Genoa, Italy, July 2017. [DOI: 10.1109/HPCS.2017.48]

(3) Hiroaki Morimoto, Khureltulga Dashdavaa, Keichi Takahashi, Yoshiyuki Kido, Susumu Date, Shinji Shimojo, "Design and Implementation of SDN-enhanced MPI Broadcast Targeting a Fat-tree Interconnect", The 2017 International Conference on High Performance Computing & Simulation (HPCS 2017), pp.252-258, Genoa, Italy, July 2017. [DOI:10.1109HPCS.2017.46]

(4) 高橋慧智，Khureltulga Dashdavaa，木戸善之，伊達 進，下條真司，"MPI 通信パターンに基づく SDN 制御を高速化するカーネルモジュールの試作と評価"，情報処理学会研究会報告，Vol．2016-OS-137，No．13，沖縄，2016 年 5 月．

(5) Baatarsuren Munkhdorj, Keichi Takahashi, Khureltulga Dashdavaa, Yasuhiro Watashiba, Yoshiyuki Kido, Susumu Date, Shinji Shimojo, "Design and Implementation of Control Sequence Generator for SDN-enhanced MPI", The 5th International Workshop on Network-aware Data Management(NDM'15), Austin, Nov. 2015. [DOI:10.1145/2832099.2832103]

(6) Keichi Takahashi, Khureltulga Dashdavaa, Baatarsuren Munkhdorj, Yoshiyuki Kido, Susumu Date, Hiroaki Yamanaka, Eiji Kawai, Shinji Shimojo, "Concept and Design of SDN-enhanced MPI Framework", The fourth edition of the European Workshop on Software Defined Networks, pp. 109-110, Sep. 2015. [DOI:10.1109/EWSDN.2015.72]

(7) Susumu Date, Hirotake Abe, Khureltulga Dashdavaa, Keichi Takahashi, Yoshuyuki Kido, Yasuhiro Watashiba, Pongsakorn U-Chupala, Kohei Ichikawa, Hiroaki Yamanaka, Eiji Kawai, Shinji Shimojo, "An Empirical Study of SDN-accelerated HPC Infrastructure for Scientific Research", International Conference Research and Innovation (ICCCRI), Singapore, Oct. 2015. [DOI:10.1109/ICCCRI.2015.13]

(8) Susumu Date, Yoshiyuki Kido, "Current Research on SDN and IoT", Southeast Asia International Joint-Research and Training Program 2015, Taichung-Keelung, Taiwan, 7-11, Dec. 2015.

(9) Khureltulga Dashdavaa, Munkhdorj Baatarsuren, Keichi Takahashi, Susumu Date, Yoshiyuki Kido, Shinji Shimojo, "A MPI Concept with Efficient Control of Network Functionality Based on SDN", PRAGMA 29, Depok, Indonesia, 7-9, Oct. 2015.(poster)

(10) Keichi Takahashi, Baatarsuren Munkhdorj, Khureltulga Dashdavaa, Susumu Date, Yoshiyuki Kido, Shinji Shimojo, "Control Sequence Generator for Generic SDN-enhanced MPI Framework", PRAGMA 28, Nara, Japan, 8-11, Apr. 2015. (poster)

(11) Keichi Takahashi, Dashdavaa Khureltulga, Yasuhiro Watashiba, Yoshiyuki Kido, Susumu Date, Shinji Shimojo, "Performance Evaluation of SDN-enhanced MPI_Allreduce on a Cluster System with Fat-tree Interconnect", The International Conference on High Performance Computing and Simulations (HPCS2014), pp. 784-792, Jul. 2014. (10.1109/HPCSim.2014.6903768)

(12) Susumu Date, "Quest for Software Defined Infrastructure for New-Generation Computing and Visualization", Southeast Asia International Joint-Research and Training Service, Dec. 2014 (invited).

(13) Susumu Date, "Software Defined Infrastructure towards HPC cloud", The 10th AEARU Workshop on Computer Science and Web Technology (CSWT-2015), p.21, Feb. 2015 (invited).

〔図書〕（計 0 件）
該当なし。

〔産業財産権〕
該当なし。

○出願状況（計 0 件）
該当なし。

○取得状況（計 0 件）
該当なし。

〔その他〕
ホームページ等

６．研究組織

(1)研究代表者

伊達　進　（DATE，Susumu）
大阪大学・サイバーメディアセンター・准教授
研究者番号：２０３４６１７５