B-021

異機種間連携ライブラリの性能評価
Development and performance evaluation of a heterogeneous coupling library

荒川 隆[†‡] 住元 真司[†] 八代 尚[§] 中島 研吾[†]
Takashi Arakawa Shinji Sumimoto Hisashi Yashiro Kengo Nakajima

1. はじめに

現代の高性能計算機はアーキテクチャの異なる複数のノード群で構成されるのが主流となっている。これは高性能計算機に求められる役割がシミュレーション(S)に加えてデータ解析(D)や機械学習(L)などに拡大したことに呼応している。ノード間連携はファイル共有で実現されることが通常であるが、この場合データの流れは一方向であり逐次的に処理が実行される。しかし異機種間でリアルタイムにデータ通信する機能があればより強固な S+D+L 連携が可能となる。このような背景に基づき h3-Open-BDEC プロジェクトの一環として異機種間通信ライブラリ h3-Open-SYS/WaitIO および連成ライブラリ h3-Open-UTIL/MP が開発されている。本講演ではこれらのライブラリの機能や構造を紹介し性能評価について報告する。

2. h3-Open-BDEC

h3-Open-BDEC は科研費基盤研究 S「（計算＋データ＋学習）融合によるエクサスケール時代の革新的シミュレーション手法」の元で開発されているソフトウェア基盤である。開発の目的はアーキテクチャの異なる複数のノード群で構成されるシステムにおいて計算＋データ＋学習が融合した処理を高性能かつ高効率に実行するためのプラットフォームを構築することである。h3-Open-BDEC は図 1 に示すように New Principle for Computations, Simulation + Data + Learning, Integration + Communication + Utilities の 3 パートに大分された 7 つのソフトウェアパッケージで構成される。本講演で紹介するのは図右端の 2 つのパッケージ h3-Open-SYS と h3-Open-UTIL で開発されたライブラリである。

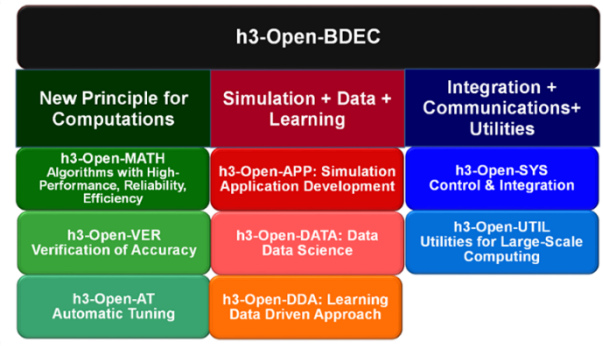

図 1 h3-Open-BDEC の構成

3. h3-Open-SYS/WaitIO

h3-Open-SYS/WaitIO(以下 WaitIO)は MPI による直接通信ができない異機種間での通信を可能とするライブラリである。WaitIO は派生的に大域通信もサポートしているが、基本的な通信機能は 1 対 1 通信であり MPI に類似のインターフェースを持つ関数 waitio_isend, waitio_irecv, waitio_wait の 3 種類の関数で通信が実現される。WaitIO はインターコネクト経由で通信を行う WaitIO-Socket, 共有ファイルシステム経由で通信を行う WaitIO-File および設定された閾値で両者を切り替える WaitIO-Hybrid の 3 種類があるが、本稿では WaitIO-Socket および WaitIO-File を使用した。ライブラリの詳細や性能については参考文献[1]および[2]で詳述されている。

4. h3-Open-UTIL/MP

h3-Open-UTIL/MP(以下 UTIL/MP)は弱連成計算のためのカップリングライブラリである。カップリングライブラリの一般的な機能は 1）複数のアプリケーションの実行管理、2）データ交換、3）異なる格子系間の補間計算の 3 つであるが、UTIL/MP はこれらの機能をシンプルなインターフェースで、かつ幅広い分野のアプリケーションに対して提供できるように設計されている。更に UTIL/MP 独自の機能として、1)アンサンブル連成機能、2）WaitIO と協調した異機種間連成機能、3)Python 連成機能、の 3 つの機能をもつ。以下では、異機種間連成機能について詳述し、性能評価の一部として Python 連成機能についても言及する。

5. 異機種間連携

5.1 UTIL/MP+WaitIO の構造

WaitIO と連携した UTIL/MP の構造を図 2 に示す。破線の四角で表されるのが一つのアプリケーションであり、図では 2 つのアプリケーションが異なるシステムで実行されている状況を表現している。図下部中央の WaitIO は WaitIO を用いた異機種間の通信を、両側の MPI は機種内(アプリケーション内)の MPI 通信を表している。最下段の MPI+WaitIO は MPI と WaitIO の両者を用いた異機種間大域通信を表す。UTIL/MP 全体は Fortran の module 群で構成されている。これらの module は大まかな階層構造を持ち、最上層には利用者が参照する API モジュールが、最下層にはプロセス管理と通信を担う MPI wrapper モジュールが配置されている。他のモジュール群は MPI wrapper モジュールを介して通信を行うようになっており、WaitIO と連携した場合は図に示したようにこのモジュールを WaitIO 対応版に変更するだけで異機種間連成が可能となる。

† 東京大学 The University of Tokyo
‡ CliMTech CliMTech Inc.
§ 国立環境研究所 National Institute for Environmental Studies

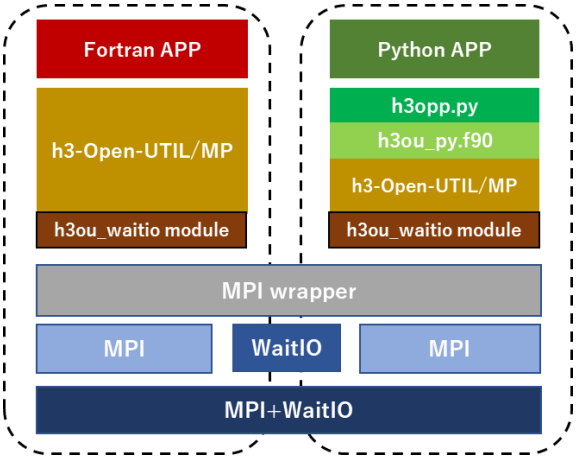

図 2　WaitIO と連携した UTIL/MP の構造

5.2 Toy モデルでの性能評価

はじめに仮想モデルを用いてデータ交換時間を測定した。UTIL/MP においてデータ交換に用いられる通信関数は waitio_isend, waitio_irecv, waitio_waito の 3 種類のみである。通信データ量を表 1 に示すように 2 倍ずつ増加させた 5 ケースについて測定を実施した。測定に用いたマシンは東京大学の Wisteria-BDEC01 の Odyssey ノードと Aquarius ノード、および名古屋大学の不老 TypeI ノードと TypeII ノードである。Odyssey と TypeI 側を 160 プロセス 40 ノードで、Aquarius と TypeII 側を 20 プロセス 1 ノードでそれぞれ実行し、Wisteria-BDEC01 では WaitIO-Socket と WaitIO-File, 不老では WaitIO-File をそれぞれ評価した。測定結果を図 3 に示す。Wisteria-BDEC01 での Socket と File を比較すると通信データ量に対して File の方が時間増加率が小さく、C2 と C3 の間で性能が逆転する。不老の時間は Wsiteria-BDEC01 のほぼ 2 倍であり、これは両者のファイルシステムの性能比をそのまま反映している。

表 1　Toy モデルでの通信データ量

Case	total size	size/process
C0	26214400	163840
C1	52428800	327680
C2	1.05E+08	655360
C3	2.1E+08	1310720
C4	4.19E+08	2621440

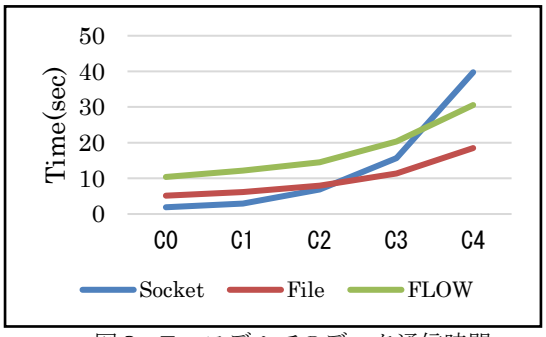

図 3　Toy モデルでのデータ通信時間

5.3 実モデルでの性能評価

評価に用いたモデルは大気モデル NICAM と機械学習ライブラリ PyTorch を用いた AI アプリケーションである。NICAM と AI の詳細は文献[3]を参照されたい。ここでは NICAM 側を水平格子数 163840, 鉛直層数 40 層、AI 側を水平格子数 10240，鉛直層数 1 としデータ交換時間を測定した。交換データ量は NICAM から AI が大半を占めるため、実質的に NICAM から AI へのデータ送受信時間が測定されることになる。通信データ総量は 1 回のデータ交換で約 368Mbyte である。NICAM を Odyssey 側 160 プロセス 40 ノードで、AI を Aquarius 側 20 プロセス 1 ノードで実行した際の、NICAM 各プロセスの通信時間を図 4 に示す。図の通り、通信時間は Odyssey 側のプロセスをどのように割り当てるかに強く影響され、プロセストポロジを指定しない場合と最適に指定した場合の通信性能はほぼ 2 倍の開きがある。最も性能のよい割り当てパタンの場合、7 回のデータ交換でおよそ 20 秒の時間を要した。これは通信速度に換算すると 128Mbyte/sec にあたる。

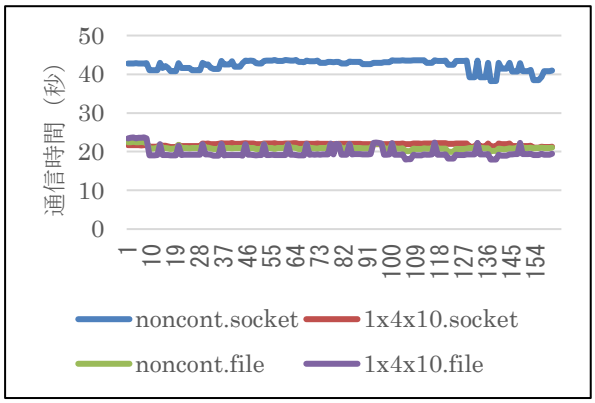

図 4　NICAM と AI 間でのデータ通信時間

6. おわりに

今後はアプリケーションの適用範囲を拡大しつつ評価を実施してゆく予定である。

謝辞

本研究の一部は科学研究費補助金（19H05662，代表：中島研吾）の助成、並びに、学際大規模情報基盤共同利用・共同研究拠点（課題番号：jh210022-MDH）の支援による。また、本研究は名古屋大学のスーパーコンピュータ「不老」の一般利用制度を利用して実施された。

参考文献

[1] Shinji Sumimoto, Takashi Arakawa, Yoshio Sakaguchi, Hiroya Matsuba, Hisashi Yashiro, Toshihiro Hanawa, Kengo Nakajima, A System-Wide Communication to Couple Multiple MPI Programs for Heterogeneous Computing. The 23rd International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT'22), (2022)

[2] 住元真司、荒川隆、坂口吉生、松葉浩也、八代尚、塙敏博、中島研吾, 情報処理学会研究報告 2022(HPC-187-6) 2022 年

[3] Takashi Arakawa, Hisashi Yashiro, Kengo Nakajima, Development of a coupler h3-Open-UTIL/MP. International Conference on High Performance Computing in Asia-Pacific Region (2022). https://doi.org/10.1145/3492805.3492809