Vol.2018-HPC-165 No.4
2018/7/30

サイエンティフィックビッグデータアプリケーションのためのベンチマークセットの提案

岩井 厚樹[1,a)]　建部 修見[2]

概要：サイエンティフィックビッグデータアプリケーションは膨大なデータを必要とするため，高性能なストレージが要求される．ストレージの IO 性能を測るにあたり数多くのアプリケーションベンチマークおよびマイクロベンチマークがある．マイクロベンチマークは汎用的な性能結果を得られるが，その性能が実際にアプリケーションを実行した時に発揮されるとは限らないという問題がある．アプリケーションベンチマークは対象のアプリケーションに特化した IO 性能を測ることができ，筑波大学計算科学研究センターではいくつかのアプリケーション IO ベンチマークを開発している．一方で，サイエンティフィックビッグデータアプリケーションは MPI だけでなく MapReduce やワークフローエンジンでも実行されている．しかし，現時点で IO 測定をするための MapReduce およびワークフローエンジンの共通のアプリケーション IO ベンチマークは提供されていない．そこで，本論文では既存のアプリケーション IO ベンチマーク，そして MapReduce およびワークフローエンジンのアプリケーション IO ベンチマークを含むベンチマークセットを提案する．これにより，異なる計算機で共通のベンチマークが実行可能となり，計算機の性能把握や予測，比較を行うことができる．評価ではいくつかのアプリケーション IO ベンチマークを実行し，IO 性能を測定することができた．

1. はじめに

気象や天文などの自然科学を解くアプリケーションは観測機器やスーパーコンピュータの発展によりデータ量が膨大となっている．例えば，気象の分野においては 30 秒間に 200 GB ものデータが生成されており [1]，天文の分野においてはハワイに設置されているすばる望遠鏡で一晩に 300 GB の観測データが生成される [2]．また，膨大なデータ量に加えて，多大なる実行時間も必要とするため様々なアルゴリズムやアーキテクチャ，ライブラリを用いた並列手法が試みられてきた．そのような膨大な実行時間を要するために並列手法を駆使し，ビッグデータを解析することで自然科学の問題を解くアプリケーションをサイエンティフィックビッグデータアプリケーションと呼ぶ．サイエンティフィックビッグデータアプリケーションは膨大なリソース，実行時間を要するため，高性能な CPU や GPU，ネットワーク機器を搭載するスーパーコンピュータで，並列且つ高速に実行されている．その際，サイエンティフィックビッグデータアプリケーションにおいては特に重要となるのが，ストレージである．先に述べた通り，データ量が膨大なため，それらを高速に処理するために高バンド幅且つ高 IOPS なストレージおよびそれと密接に関係するファイルシステムが必要とされる．

一方で，昨今のストレージは CPU やメモリの実行速度に比べると格段に遅いという問題がある．そのため，ストレージ性能を向上させる研究が活発に行われており，その際重要な指標となるのがベンチマークの結果である．ベンチマークを行うことで評価対象の性能値の把握および予測，改善点の発見をすることができる．ストレージおよびファイルシステムの分野においては，ベンチマークの結果としてバンド幅および IOPS が注目される．最近では IO500[3] というスーパーコンピュータの IO 性能を競うリストも新設されるほど，スーパーコンピュータにおいて IO 性能は重要視されている．

IO を測定するベンチマークは数多くあるが，大きく分けると 2 種類に区別できる．それはマイクロベンチマークとアプリケーションベンチマークである．マイクロベンチマークはアプリケーションのある一部分の性能を測定する際に用いられる．例えば，シーケンシャルリードの性能測定や，転送サイズが N MB でのライト性能のような評価対象が単純な場合に活用される．しかし，マイクロベンチマークで得られた結果が，必ずしもアプリケーションを実

1 筑波大学大学院システム情報工学研究科コンピュータサイエンス専攻
2 筑波大学計算科学研究センター
a) iwai@hpcs.cs.tsukuba.ac.jp

行した際に発揮できるとは限らないという問題がある．アプリケーションベンチマークはアプリケーションの全体もしくはその一部を用いて性能評価を行うため，確かな実行結果を得ることができる．ゆえに，IO に関して言えば，アプリケーションの IO 部分だけを抜き出したベンチマークを開発すれば，そのアプリケーションを実機で実行する際の IO の見積もりを行うことができる．そのような IO アプリケーションベンチマークは，IO が頻繁且つ膨大なデータがやりとりされるアプリケーションにとって需要がある．

また, サイエンティフィックアプリケーションは MPI だけではなく MapReduce[4] や, Pegasus[5], Swift[6], Pwrake[7] などのワークフローエンジンでも実行されている．しかし，それらのベンチマークは未だ共通のベンチマークソフトが確立されていないという課題がある．

そこで本研究では，いくつかのアプリケーション IO ベンチマークおよび MapReduce，ワークフローエンジンのベンチマークソフトを一つのパッケージにまとめて提供し，性能評価を複数の異なるマシンで評価可能とするベンチマークセットを提案する．これにより，今まで評価されていなかった MapReduce，ワークフローエンジンの性能評価が可能になるに加え，いくつかのサイエンティフィックビッグデータアプリケーションを異なるスーパーコンピュータで実行する際の IO の性能評価をすることが可能となる．

以下 2 章では，HPC で用いられるベンチマークソフトについて述べる．3 章では，HPC におけるベンチマークの現状およびそれを踏まえてたベンチマークセットの提案を行う．4 章では，ベンチマークセットの一部を用いたスーパーコンピュータのストレージ性能の評価およびその結果について述べる．5 章では，まとめと今後の課題について述べる．

2. 関連技術

本章では HPC で用いられるいくつかのベンチマークソフトについて述べる．

2.1 IOR，mdtest，FIO

IOR，mdtest，FIO は HPC においてストレージ性能を評価する際にしばしば用いられるマイクロベンチマークである．

IOR[8] はバンド幅を測定するためのベンチマークソフトである．IOR はオプションを設定することで POSIX，MPI-IO などの複数のインターフェース，共有ファイル書き込み，Direct-IO など様々な設定をすることができ，アプリケーションの IO を再現することができる．IOR は後述する mdtest と共に IO500 の評価ソフトとして採用されており，HPC 界における最も代表的なバンド幅測定ベンチマークソフトと言える．

mdtest[9] は IOPS を測定するためのベンチマークソフトである．ファイルおよびディレクトリの create，remove，stat を元に IOPS 値を測る．MPI に対応しているため，複数ノードでの実行も可能となっている．

FIO（Flexible I/O Tester）[10] はストレージのバンド幅を測定するベンチマークツールであり，POSIX スレッドを用いて並列実行することができる．

いずれのベンチマークも MPI や POSIX スレッドを用いることで複数ノード環境での並列ベンチマークが可能であるが，MapReduce およびワークフロー系には対応しておらず，それらを用いたアプリケーションを実行した際の性能を直接見積もれるとは言えない．

2.2 Mantevo，NPB

Mantevo および NPB（NAS Parallel Benchmarks）は既存のベンチマークセットである．

Mantevo[11] は Miniapp という複数のアプリケーションの集合であり，それらのアプリケーションはサイエンティフィックアプリケーションでしばしば使われる代表的な物理モデルを含んでいる．物理モデルは数学的には異なるが，プログラムにおいては非常に似たコードの特徴を持っているため，異なるアプリケーションであっても物理モデルが似ている場合は，Miniapp を用いてある程度の性能予測をすることが可能である．

NAS Parallel Benchmarks[12] とは，アメリカ航空宇宙局 NASA が開発するベンチマークツールであり，5 つの並列カーネルベンチマークと 3 つのシミュレーションアプリケーションベンチマークから構成される．シミュレーションアプリケーションベンチマークには，ブロックサイズ 5 x 5 を block Tridiagonal solver で解く BT（Block Tridiagonal），scalar pentadiagonal を解く SP（Scalar Pentadiagonal Solver），放物型および楕円型偏微分方程式を解くための LU 分解を行う LU（Luwer Upper Gauss - Sidel Solver）が含まれる．

いずれも複数のアプリケーションベンチマークを含むベンチマークセットであるが，IO 性能測定には主眼を置いていない．

3. ベンチマークセットの提案

ベンチマークは計算機のパフォーマンスの把握および予測をするうえで非常に重要である．サイエンティフィックビッグデータアプリケーションでは，データ量が膨大なためストレージ性能が重要視されており，単位時間あたりにどれほどのデータを転送できるかを表すバンド幅および単位時間あたりにどれほどの IO 操作ができるかを表す IOPS が注目される．それらを測るツールとして，IO500 にも採用されている IOR および mdtest がしばしば用いられる．IOR および mdtest はある限られた状況を想定して特定の操作のみを評価するマイクロベンチマークに区分される．

マイクロベンチマークは特定の評価に特化しているため汎用的な評価を得ることができるが，その性能が実際にアプリケーションを実行した時に発揮されるかは定かでない．そこで，実際にアプリケーションを実行して性能評価をするプログラムとしてアプリケーションベンチマークがある．実際にアプリケーションを実行するため，評価したい箇所の実測値を得ることができる．しかし，アプリケーションの全ステップを実行するには多大な時間と膨大なリソースを必要とする．そこで，アプリケーションの一部を抜き出して特定の処理の性能を測ることもある．サイエンティフィックビッグデータアプリケーションに関して言えば，IO 部分のみを抜粋 IO 性能を測定することが考えられる．

アプリケーションの IO 部分のみを評価するベンチマークソフトとして，ARGOT-IO，LES-IO，QCD-IO が筑波大学計算科学研究センターにて研究開発されている．ARGOT-IO は輻射輸送シミュレーションプログラムである ARGOT[13] の，メッシュ情報を書き出す部分を抜き出した IO ベンチマークである．LES-IO は筑波大学日下研究室で開発されている都市部の気象現象を対象にした Large Eddy Simulation（LES）モデル [14] の IO 部分を抜き出したベンチマークソフトである．LES は解像度が高いため都市内部で発生する複雑な乱流構造を計算することができるが，グリッド数およびデータ量が増えやすい．QCD-IO は，格子 QCD シミュレーションにおいて最も計算量を必要とする quark propagators を解析する処理により CPU および MPI の性能を評価することを目的に開発された CCS QCD Benchmark[15] の，設定ファイルの読み書きにより IO 性能を測定するベンチマークである．

先にあげたアプリケーションベンチマークは全て MPI を用いて並列実行されているが，サイエンティフィックビッグデータアプリケーションは MapReduce やワークフローエンジンでも実行されている．MapReduce は分散処理を可能とするプログラミングモデルの一つであり，耐障害性を持ちながらスケーラブルにビッグデータ処理を行うことができる．MapReduce を HPC に適用する研究は既に行われている [16][17]．MapReduce を評価するベンチマークとして MRBench[18] があるが，これは分析および集計のためのクエリの性能評価を行う TPC-H[19] を対象としているため，サイエンティフィックビッグデータアプリケーションのベンチマークには適用できない．そのため，スーパーコンピュータ上で MapReduce を用いてサイエンティフィックビッグデータアプリケーションを実行した際の IO 性能を評価するような共通のベンチマークが求められる．また，ワークフローエンジンも Pegasus や Swift，Pwrake などがあるが，それらをスーパーコンピュータ上で実行した際にどれほどの IO 性能を発揮するかを評価するベンチマークも提供されていない．

HPC において，MapReduce とワークフローエンジンのための共通のベンチマークは提供されていないため，研究者は自作のベンチマークソフトを使わなければならず，他研究との比較ができないという問題が発生する．そこで，MapReduce およびワークフローエンジンのためのベンチマーク，そして先にあげた ARGOT-IO，LES-IO，QCD-IO およびその他のアプリケーション IO ベンチマークを含むベンチマークセットを提案する．これにより，これまでできていなかった MapReduce およびワークフローエンジンを用いた IO 性能評価が可能になる．また，ARGOT-IO などのアプリケーション IO ベンチマークにより対象のサイエンティフィックアプリケーションの IO 性能を測ることができる．そして，ベンチマークセットを配布することにより異なる計算機で実行した場合の性能比較も可能となる．

表 1　評価環境

CPU	Intel Xeon Phi 7250
コア数	68
メモリ	16 GB MCDRAM，96 GB DDR4-2400
OS	CentOS 7
ネットワーク	Intel Omni-Path
ファイルシステム	DDN Lustre

3.1 MapReduce およびワークフローエンジンのアプリケーション IO ベンチマーク

MapReduce およびワークフローエンジンを用いてサイエンティフィックビッグデータアプリケーションを実行する研究は既に行われいてる．MapReduce においては，GHOST-MP というメタゲノムアプリケーションを MapReduce に適用した事例がある [20]．ワークフローエンジン Pwrake では，天体のモザイク画を生成する Montage[21] や，ゲノム解析を行う GHOST-Z[22] というアプリケーションを Pwrake を用いて実行する研究が行われている [23][24]．これらのアプリケーションの IO 部分を抜粋することで MapReduce およびワークフローエンジンを用いたアプリケーション IO ベンチマークを実現できる．また，ワークフローエンジンに関しては，他のワークフローエンジンでも同じアプリケーションのワークフローを記述し実行することで，異なるワークフローエンジンとの比較が可能となる．

4. 評価

本研究では JCAHPC が運営する Oakforest-PACS（以下 OFP）にて，ARGOT-IO，LES-IO，QCD-IO を用いて IO 性能を測定した．

4.1 評価環境

評価環境である OFP のシステム構成を**表 1** に示す．ベンチマークを実行する際には Flat モードで行った．

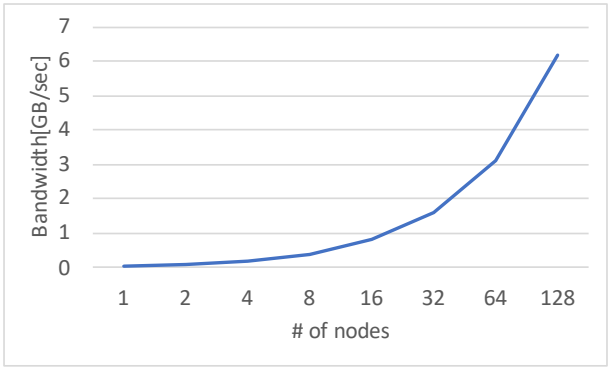

図 1　ARGOT-IO によるバンド幅評価

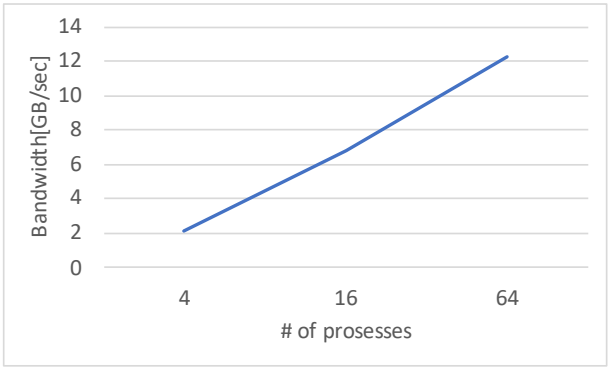

図 2　LES-IO による単一ノードでのバンド幅評価

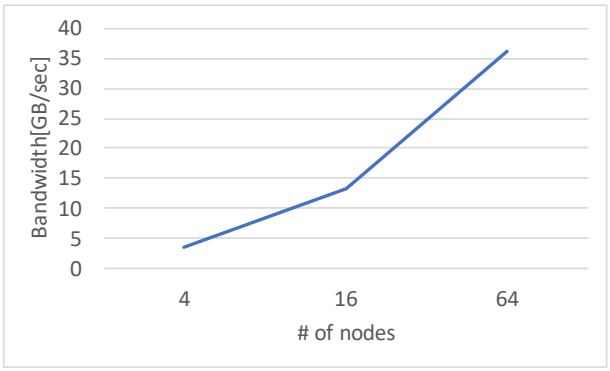

図 3　LES-IO による複数ノードでのバンド幅評価

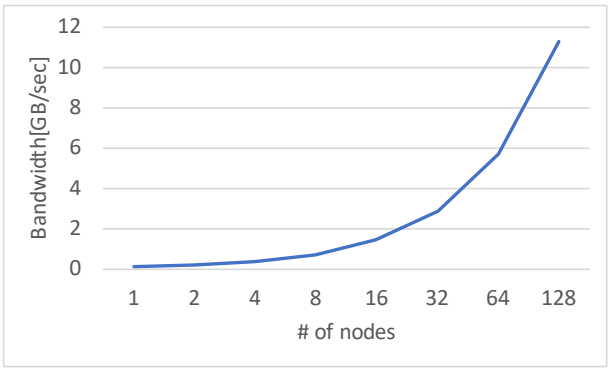

図 4　QCD-IO によるバンド幅評価

4.2　ARGOT-IO

ARGOT-IO はシミュレーションプログラム ARGOT のメッシュ情報を書き出す部分を抜き出したベンチマークプログラムである．ベンチマークプログラムを実行するにあたり，メッシュサイズとプロセス数を指定する必要がある．メッシュサイズとプロセス数は X 軸， Y 軸， Z 軸の 3 軸それぞれ指定する必要がある．本評価では 1 プロセスあたりのメッシュサイズを 256 とした．プロセス数は，3 軸のプロセス数の乗算値が 1 から 2 冪で 128 まで変化するように設定した．この時，1 ノードあたりのプロセス数は 1 としたため，最大で 128 ノード使用した．評価結果を**図 1** に示す．ノード数を増やしてもバンド幅はスケールし，128 ノードで 6 GB/sec という結果を得られた．

4.3　LES-IO

LES-IO を OFP にて実行した結果を**図 ??**および**図 ??**に示す．LES は Unidata の netCDF[25] を利用して IO を発行しており，さらに HDF5[26] の MPI-IO を用いて並列 IO を行なっており，LES-IO ではその並列 IO の部分を抜き出している．LES-IO は，シミュレーションの積算値の出力（Integral（Write）），シミュレーションの瞬間値（Instant（WRITE），シミュレーションの平均値の出力（Average（WRITE），シミュレーションのリスタートに必要なデータの出力（Restart（WRITE）），リスタートファイルの読み込み（Restart（READ））の 5 つのファイル IO を行う．本評価では Instant（WRITE）の評価を行なった．メッシュサイズを 256 × 256 × 512 に固定し，図 **??**ではシングルノード実行で MPI のプロセス数を 4, 16, 64, 256，図 **??**では 1 ノードあたりのプロセス数は 1 としノード数を 4, 16, 64 のように変化させながら実行した．シングルノード実行では，64 プロセスで 12 GB/sec という結果を得られた．複数ノード実行では，ノード数を増やしてもバンド幅は向上し，64 ノードで約 35 GB/sec という結果を得られた．

4.4　QCD-IO

QCD-IO では CCS-QCD の設定ファイルの読み書きにより IO を評価している．ベンチマークを実行するには問題サイズとプロセス数を指定する必要がある．1 プロセスあたりの問題サイズは，X 軸は 12，Y 軸および Z 軸は 24 とした．本評価では 1 ノードあたりのプロセス数を 4 とし，ノード数を 1 から 2 冪で 128 まで変化させながら書き込みのバンド幅評価を行なった．この時，ノード数 128 で総プロセス数は 512 となり，総書き込みサイズは 2TB ほどである．評価結果を**図 4** に示す．ノード数を増やしてもバンド幅は増加し，128 ノードで約 11 GB/sec という結果が得られた．

5.　まとめ

本論文ではサイエンティフィックビッグデータアプリケーションを想定したベンチマークセットの提案およびそ

れを用いた評価を行なった．評価では ARGOT-IO，LES-IO，QCD-IO を OFP にて実行し，それらのアプリケーションを実行した際の IO 性能を見積もることができた．

今後の課題として，MapReduce およびワークフローエンジンのためのアプリケーション IO ベンチマークの設計，開発，評価をする必要がある．

謝辞 本研究の一部は，JST-CREST ACA20935，JSPS 科研費 16K16108，JST-CREST JPMJCR1303，JST-CREST JPMJCR1413，JSPS 科研費 JP17H01748 による．

参考文献

[1] Matsuoka, S., Sato, H., Tatebe, O., Koibuchi, M., Fujiwara, I., Suzuki, S., Kakuta, M., Ishida, T., Akiyama, Y., Suzumura, T. et al.: Extreme Big Data (EBD): Next Generation Big Data Infrastructure Technologies Towards Yottabyte/Year, *Supercomputing frontiers and innovations*, Vol. 1, No. 2, pp. 89–107 (2014).

[2] 田中昌宏，建部修見，川島英之：すばる HSC パイプラインの Pwrake/Gfarm による高速化手法の提案，研究報告ハイパフォーマンスコンピューティング (HPC)，Vol. 2017-HPC-162, No. 9, pp. 1–8 (2017).

[3] IO500, https://www.vi4io.org/.

[4] Dean, J. and Ghemawat, S.: MapReduce: Simplified Data Processing on Large Clusters, *Communications of the ACM*, Vol. 51, No. 1, pp. 107–113 (2008).

[5] Deelman, E., Vahi, K., Juve, G., Rynge, M., Callaghan, S., Maechling, P. J., Mayani, R., Chen, W., da Silva, R. F., Livny, M. et al.: Pegasus, a workflow management system for science automation, *Future Generation Computer Systems*, Vol. 46, pp. 17–35 (2015).

[6] Zhao, Y., Hategan, M., Clifford, B., Foster, I., Von Laszewski, G., Nefedova, V., Raicu, I., Stef-Praun, T. and Wilde, M.: Swift: Fast, Reliable, Loosely Coupled Parallel Computation, *Services, 2007 IEEE Congress on*, IEEE, pp. 199–206 (2007).

[7] Tanaka, M. and Tatebe, O.: Pwrake: a parallel and distributed flexible workflow management tool for wide-area data intensive computing, *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, ACM, pp. 356–359 (2010).

[8] IOR, https://github.com/LLNL/ior.

[9] mdtest, https://sourceforge.net/projects/mdtest/.

[10] Flexible I/O Tester, https://github.com/axboe/fio.

[11] Heroux, M. A., Doerfler, D. W., Crozier, P. S., Willenbring, J. M., Edwards, H. C., Williams, A., Rajan, M., Keiter, E. R., Thornquist, H. K. and Numrich, R. W.: Improving Performance via Mini-applications, *Sandia National Laboratories, Tech. Rep. SAND2009-5574*, Vol. 3 (2009).

[12] Bailey, D. H., Barszcz, E., Barton, J. T., Browning, D. S., Carter, R. L., Dagum, L., Fatoohi, R. A., Frederickson, P. O., Lasinski, T. A., Schreiber, R. S. et al.: The NAS Parallel Benchmarks, *NAS Techinical Report RNR-94-007*, pp. 1–79 (1994).

[13] Okamoto, T., Yoshikawa, K. and Umemura, M.: ARGOT: accelerated radiative transfer on grids using oct-tree, *Monthly Notices of the Royal Astronomical Society*, Vol. 419, No. 4, pp. 2855–2866 (2012).

[14] 筑波大学日下博幸研究室，http://www.geoenv.tsukuba.ac.jp/ kusakaken/index.php?id=5.

[15] Boku, T., Ishikawa, K.-I., Kuramashi, Y., Meadows, L., DMello, M., Troute, M. and Vemuri, R.: A performance evaluation of CCS QCD Benchmark on the COMA (Intel (R) Xeon Phi ˆ{*TM*}, KNC) system, *arXiv preprint arXiv:1612.06556* (2016).

[16] Ekanayake, J., Pallickara, S. and Fox, G.: Mapreduce for Data Intensive Scientific Analyses, *eScience, 2008. eScience'08. IEEE Fourth International Conference on*, IEEE, pp. 277–284 (2008).

[17] Fadika, Z., Dede, E., Govindaraju, M. and Ramakrishnan, L.: MARIANE: MApReduce Implementation Adapted for HPC Environments, *Proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing*, IEEE Computer Society, pp. 82–89 (2011).

[18] Kim, K., Jeon, K., Han, H., Kim, S.-g., Jung, H. and Yeom, H. Y.: MRbench: A benchmark for mapreduce framework, *Parallel and Distributed Systems, 2008. ICPADS'08. 14th IEEE International Conference on*, IEEE, pp. 11–18 (2008).

[19] TPC-H, http://www.tpc.org/tpch/.

[20] Matsuda, M., Maruyama, N. and Takizawa, S.: K MapReduce: A Scalable Tool for Data-Processing and Search/Ensemble Applications on Large-Scale Supercomputers, *Cluster Computing (CLUSTER), 2013 IEEE International Conference on*, IEEE, pp. 1–8 (2013).

[21] Montage, http://montage.ipac.caltech.edu/.

[22] Suzuki, S., Kakuta, M., Ishida, T. and Akiyama, Y.: Faster sequence homology searches by clustering subsequences, *Bioinformatics*, Vol. 31, No. 8, pp. 1183–1190 (2014).

[23] 田中昌宏，建部修見：並列分散ワークフローシステム Pwrake による大規模データ処理 (宇宙科学情報解析論文誌 第一号)，宇宙航空研究開発機構研究開発報告，Vol. 11, pp. 67–75 (2012).

[24] 町田健太，建部修見：Pwrake/Gfarm による分散並列相同性検索システムの提案，研究報告ハイパフォーマンスコンピューティング (HPC)， Vol. 2017-HPC-162, No. 10 (2017).

[25] Network Common Data Form(NetCDF), https://www.unidata.ucar.edu/software/netcdf/.

[26] HDF5, https://support.hdfgroup.org/HDF5/.