MapReduceにおけるファイルシステムの性能評価

三上　俊輔 † 建部　修見 ††
† 筑波大学第三学群情報学類　†† 筑波大学大学院システム情報工学研究科

1　はじめに

MapReduce[1] とは大規模データを並列分散処理するためのフレームワークである．近年，データが大規模化し，そのデータを妥当な時間で処理するためには数百台のマシンで分散処理をする必要がある．効率よく分散処理するには並列計算，エラー処理，データ分散など複雑な処理が必要となる．MapReduce のモデルを使えば，そういった複雑な処理を MapReduce が行い，プログラマは実際の計算のための処理だけを記述して分散処理を実行できる．

この MapReduce は一つのクラスタ内で利用されることを想定しているが，科学研究分野では実験設備などの理由から距離的に離れたストレージやクラスタにデータが置かれることが多く，そのような環境を広域環境という．本研究では広域環境において性能評価を行い，MapReduce を利用するための問題点を検討する．さらに，広域環境を想定した分散ファイルシステムとして Gfarm[3] があり，すでに多くの研究機関で利用されている．この Gfarm 上のデータに対して効率良く分散処理をしたいという要求もあり，Gfarm 上での MapReduce の処理の性能評価を行った．

2　Hadoop

Hadoop[2] は Google の MapReduce のオープンソース実装で，現在 Apache Software Foundation によりオープンソースで開発が進められている．Hadoop での計算タスクは HDFS という分散共有ファイルシステム上で Hadoop MapReduce という分散処理システムが動作することで実行される．

2.1　HDFS

HDFS は Hadoop で標準的に使われている分散共有ファイルシステムである．NameNode と呼ばれるメタデータサーバがファイルシステムのメタデータを管理し，DataNode と呼ばれるデータサーバが実際のデータを保持する．データは一定のサイズのブロック (デフォルトは 64MB) に分割され，各スレーブノードのローカルストレージに分散されて保存されるが，一つのファイルシステムとしてみることができる．

2.2　Hadoop MapReduce

MapReduce のプログラミングモデルでは処理を Map，Shuffle，Reduce の各フェーズに分解する．Map では key/value のペアを入力データとして受け取り，ユーザー定義の map 処理を実行し，中間 key/value を生成する．Shuffle では，中間 key/value を受け取り同じ key に対して value のリストを生成し，ソートして Reduce に渡す．Reduce では，key と対応する value のリストを受け取り，ユーザー定義の reduce 処理を行い，最終出力となる key/value データを生成する．

JobTarcker と呼ばれるマスタがこの一連の MapReduce ジョブを管理し，TaskTracker と呼ばれるワーカーが実際の Map タスクと Reduce タスクを実行する．

TaskTracker と DataNode は同じノードで実行されるため (図 1)，タスクをデータの近くに割り当てることにより，ネットワークのデータ転送量が抑えられ，効率的な I/O を実現することができる．

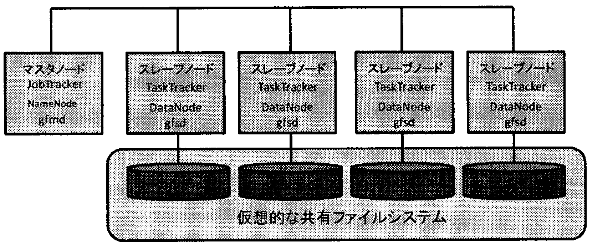

図 1: Hadoop MapReduce，Hadoop-Gfarm の構成図

3　Hadoop-Gfarm

Gfarm は広域分散ファイルシステムで，メタデータサーバ gfmd がファイルシステムのメタデータを管理し，gfsd が実際のデータを保持する．Hadoop では Hadoop からそのファイルシステムを利用する API(open や read など) を定義すれば HDFS 以外の共有ファイルシステムも利用可能である．すでに Hadoop-Gfarm[4] プラグインが開発されており，Hadoop のファイルシステムとして Gfarm を利用できる．今回は gfsd を Hadoop の TaskTracker と同じノードで実行しているため (図 1)，HDFS 同様，タスクをデータの近くに割り当てネットワークのデータ転送量を抑えることが可能である．効率的な I/O の実現のためには，この機能は重要であることが考えられ，

Performance Evaluation for FileSystem on MapReduce
† Shunsuke Mikami
†† Osamu Tatebe
College of Information Sciences, Third Cluster of Collleges, University of Tsukuba (†)
Graduate School of Systems and Information Engineering, University of Tsukuba (††)

本研究開発でその機能を実現した．具体的には Gfarm のファイルの複製のあるノードの一覧を返す関数を実装した．その関数を実装することにより，データの配置を考慮したタスクのスケジューリングを行うことが可能となる．

4 性能評価

HDFS と Gfarm の性能を比較する環境として InTrigger[5] を使用した．InTrigger は日本国内 14 拠点，数百ノードで構成される分散処理の研究などのためのプラットフォームである．今回は 1 拠点のみ使用し，各ノードは 2 ソケット 4 コア Intel Xeon E5410 (2.33GHz) で構成され，メモリ容量は 32GB である．

4.1 書き込み性能

図 2 にノード数を変えて各ノードで 5GB のファイルを MapReduce のジョブとして同時に書き込んだ時の HDFS と Gfarm の書き込み性能を示す．理想曲線は 1 ノードで HDFS へ書き込んだ時の性能，約 60MB/sec を基準にしている．Gfarm が性能的に若干劣っているようだが，HDFS の場合はメモリから HDD に完全に書き出す前にジョブが終了しているのに対して，Gfarm は完全に HDD に書き込んでから終了している．実際に HDFS に 2 回連続で書き込みを行った場合は 2 回目が遅くなり，この時の速度は Gfarm と同程度であった．本質的な性能の差はないと言える．

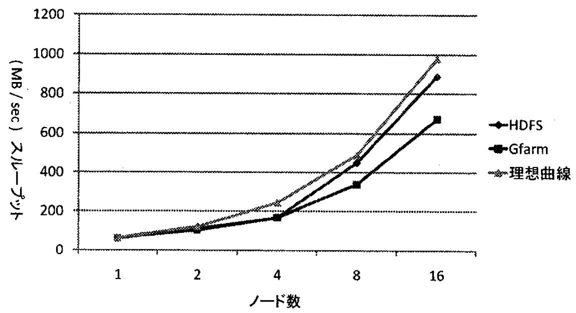

図 2: 書き込み性能の比較

4.2 読み込み性能

図 3 に書き込み性能の測定時に書き込んだ各ノードで 5GB のファイルを MapReduce のジョブとして同時に読み込んだ時の性能を示す．理想曲線は 1 ノードで Gfarm へ読み込んだ時の性能，約 35MB/sec を基準にしている．メモリから読み込まないように，書込み後ディスクのバッファキャッシュをフラッシュしてから計測した．Gfarm w/ affinity は本研究で実装したデータ配置を考慮したスケジューリングを可能とした結果で，Gfarm w/o affinity はデータ配置と無関係にスケジューリングした結果である．ノード数が少ない時は性能差があまりないのに対してノード数が増えると性能差が出てきて，16 ノードの場合は約 50%の性能向上がみられた．今回は 1 クラスタ内での測定であったが，ラックをまたぐ環境や広域環境ではこの差は広がると考えられる．

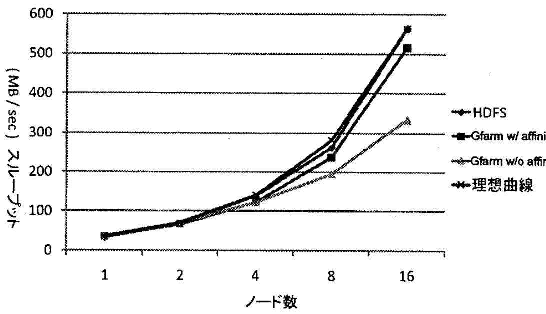

図 3: 読み込み性能の比較

5 まとめと今後の課題

HDFS と Gfarm は小規模クラスタ環境では読み込み，書き込み共に同程度の性能を持つことがわかった．HDFS に対して Gfarm は広域分散環境における認証機構や，マウントしてアクセス可能などのメリットがあり，Gfarm を Hadoop のファイルシステムとして利用することは有望であることがわかった．今後の課題として，ノード数を数百，数千と増やした時の性能，ノードが広域にまたがる場合の性能，さらに書き込み，読み込みだけの性能ではなく，実際のアプリケーションを動かした時の性能の評価も行っていきたい．

謝辞 本研究の一部は，文部科学省科学研究費補助金特定領域研究課題番号 21013005 および文部科学省次世代 IT 基盤構築のための研究開発「e-サイエンス実現のためのシステム統合・連携ソフトウェアの研究開発」，研究コミュニティ形成のための資源連携技術に関する研究 (データ共有技術に関する研究) による．

参考文献

[1] Jeffrey Dean, Sanjay Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. OSDI'04: San Francisco, CA, December, 2004.

[2] Apache Hadoop http://hadoop.apache.org/

[3] 建部修見, 曽田哲之. 広域分散ファイルシステム Gfarm v2 の実装と評価. 情報処理学会研究報告, 2007-HPC-113, pp.7-12, 2007 年 12 月

[4] Hadoop-Gfarm https://gfarm.svn.sourceforge.net/svnroot/gfarm/gfarm_hadoop/trunk/

[5] 斎藤秀雄, 鴨志田良和, 澤井省吾, 弘中健, 高橋慧, 関谷岳史, 頓楠, 柴田剛志, 横山大作, 田浦健次朗 InTrigger：柔軟な構成変化を考慮した多拠点に渡る分散計算機環境情報処理学会研究報告 HPC-111 (SWoPP 2007), 旭川 (2007)