Vol.2014-ARC-210 No.2
Vol.2014-OS-129 No.2
2014/5/14

Hadoopクラスタの動的構成変更による低電力化手法の提案

小野 貴継[1]　谷本 輝夫[1]　三吉 貴史[1]

概要：
Hadoop 上で実行される機械学習プログラムには，CPU 性能やメモリ容量を必要としない Transfer Phase と，高い CPU 性能と大容量のメモリを必要とする Analysis Phase がある．Transfer Phase は低電力な CPU を搭載したサーバで実行し，Analysis Phase は高性能な CPU を搭載したサーバで実行することで消費電力を削減する手法を提案する．フェーズごとに構成が異なるサーバで実行するためには，サーバ間でデータを移動する必要がある．本稿では，著者らが開発を進めている Disk Area Network（DAN）を用いてデータの移動を実現する．DAN を介して HDD を低電力サーバへ接続しておき，Transfer Phase を実行する．次に DAN の機能を利用して低電力サーバから HDD の接続を解除し，高性能サーバへ接続して Analysis Phase を実行する．本手法により，高性能サーバのみを用いてプログラムを実行した場合と比較して実行時間の増加は 4.3%に抑制しつつ，Transfer Phase の消費電力を約 28%削減可能であることを確認した．

1. はじめに

データセンタでは Total Cost of Ownership（TCO）の削減に対する要求が強い．TCO は主に Capital expenses（CAPEX）と Operational expenses（OPEX）の和で表すことができる [3]．OPEX はデータセンタ運用に関わるコストであり，保守管理費用やデータセンタで消費する電力の費用などが含まれる．サーバの消費電力削減は OPEX の削減につながることから，データセンタ事業者にとって重要な課題である．

データセンタでは仮想サーバによるサービスの提供が普及している．一方で，多くの物理サーバを必要とするアプリケーションも実行される．そのひとつとして，データ解析アプリケーション・プログラムが挙げられる．このようなアプリケーションでは，MapReduce [4] などを用いて複数のサーバで分散処理することで，高速な処理を実現している．本稿では機械学習アプリケーション・プログラムのひとつである Apache Mahout [2]（以下，Mahout）を用いる．Mahout は Apache Hadoop [1]（以下，Hadoop）上で実行される．一般に，Hadoop は複数台の物理サーバに処理を分散する．Hadoop では，Storage Area Netowrk（SAN）接続などによる共有ストレージは用いない．物理サーバに内蔵された HDD に分散してデータを保持し，Hadoop Distributed File System（HDFS）と呼ばれる分散共有ストレージを構成する．

Hadoop クラスタ上で Mahout を実行し特徴を解析した．CPU 性能やメモリ容量を必要としないフェーズと，高い CPU 性能と大容量のメモリを必要とするフェーズがあることが分かった．前者のフェーズはデータの移動が主な処理であることから Transfer Phase と呼び，後者はデータを解析するフェーズであることから Analysis Phase と呼ぶ．Transfer Phase はディスク I/O 性能が重要であり，CPU 性能やメモリ容量は実行時間短縮に大きく寄与しない．一方，Analysis Phase は高い CPU 性能とメモリ容量により性能向上を図ることができる．したがって，Transfer Phase と Analysis Phase とでハードウェアに対する要求が異なる Mahout を高速に実行するためには，高い CPU 性能と大容量メモリを搭載したサーバで実行する必要がある．しかしながら，Transfer Phase では高性能な CPU と大容量のメモリは実行時間の短縮に貢献しないことから不要である．これは，サーバ資源の利用効率の低下を招き，電力あたりの性能が低下するという問題が生じる．

この問題を解決するため，サーバ資源の利用効率低下を抑制することが考えられる．ジョブを割り当てるスケジューラを工夫することによって，利用効率を改善する研究が進められている [5]．サーバ資源は CPU やメモリ，ストレージ性能やネットワーク性能等複数のパラメータから

[1] 株式会社富士通研究所

なり，スケジューリングによりそれらすべての利用効率向上は難しい．さらに，一度割り当てを決めるとプログラムの振る舞いの変化に追従できない．したがって，ジョブを実行するサーバ自体の構成を変更可能にし，プログラムの特性に合ったサーバで実行することが望ましい．また，低負荷時の消費電力を削減することでエネルギー・プロポーショナリティを改善することも考えられる．サーバコンポーネント単位の改善 [6] だけでなく，サーバアーキテクチャの観点から改善する手法も提案されている [14]．このような手法では既存のサーバアーキテクチャを変更する必要があるため，新しいアーキテクチャ上でソフトウェアの検証などを再度実施しなければならない．本稿では，ソフトウェアへの影響なしにサーバアーキテクチャレベルで消費電力を改善することを目指す．

Mahout の特徴解析の結果に基づき，フェーズごとに異なるサーバ構成で実行することで消費電力を削減する手法を提案する．Transfer Phase は低電力 CPU を搭載したサーバ（低電力サーバ）で実行し，Analysis Phase は高性能な CPU を搭載したサーバ（高性能サーバ）で実行する．このとき，低電力サーバに内蔵された HDD 上にあるデータを，高性能サーバに移動させる必要がある．この問題を解決するために，我々が開発を進めている Disk Area Network（DAN）を用いた．DAN はサーバに内蔵される HDD と同様に，SAS/SATA のプロトコルを利用可能であり，性能も同等である．低電力サーバに DAN スイッチを介して HDD を接続し，Transfer Phase を実行する．Transfer Phase の実行が完了した後，DAN スイッチの機能を利用して低電力サーバから HDD を切断し，当該 HDD を高性能サーバに接続する．高性能サーバで Analysis Phase を実行し，プログラムの実行を完了する．このように，DAN スイッチにより低電力サーバの HDD 高性能サーバ接続することで，データの移動を実現する．高性能サーバのみを用いてプログラムを実行した場合と比較して，実行時間の増加は 4.3%に抑制しつつ，Transfer Phase の消費電力を約 28%削減可能であることを確認した．

本稿の構成は以下のとおりである．第 2 章で Mahout の処理について説明し，性能とハードウェアの消費電力の観点から特徴を解析する．特徴解析の結果から，DAN を用いた解決策について第 3 章で述べる．第 4 章では提案手法の評価を行い有効性を明らかにする．第 5 章で関連研究について述べ，第 6 章でまとめる．

2. Hadoop クラスタ上で動作する機械学習プログラムの特徴

2.1 Apache Mahout の動作

Mahout は Hadoop フレームワーク上で分散して実行することが可能である．Mahout には分類器，レコメンデーション，クラスタリングアルゴリズムが実装されている [12]．本稿ではこれらのアルゴリズムの中から分類器を対象に議論を進める．Mahout の分類器は次の手順で実行する．

(1) 分類器生成のために用いるデータを 64MB 単位に分割し HDFS 上に配置．
(2) カテゴリリストを用いて入力データを準備．
(3) トレーニング用のデータを使って分類器の生成．
(4) テスト用のデータを用いて分類器の精度をテスト．

本稿では（1）の処理を Transfer Phase，（2）～（4）の処理を Analylsis Phase と呼ぶ．

Hadoop で処理するため分類器生成に用いるデータを HDFS 上に配置する必要がある．これは Mahout だけでなく，Hadoop 上で実行するためには必要な処理である．トレーニングデータとテストデータを対象にカテゴリリストを用いて分類器生成およびテストのための入力データを準備する．トレーニング用入力データを用いて分類器のモデルを生成し，テスト用入力データによって分類器をテストする．

2.2 Mahout の性能と Hadoop クラスタの消費電力

Mahout の実行時間とサーバの消費電力を調査するため，次の 2 つのクラスタを用いる．

- **高性能 Hadoop クラスタ**：1 台の NameNode，3 台の DataNode で構成され，このうち DataNode はすべて高性能な CPU と大容量のメモリを搭載した高性能サーバ（HPS: High Performance Server）で構成される．
- **低電力 Hadoop クラスタ**：1 台の NameNode，3 台の DataNode で構成され，このうち DataNode はすべて低電力な CPU と小容量のメモリを搭載した低電力サーバ（LPS: Low Power Server）で構成される．

高性能 Hadoop クラスタと低電力 Hadoop クラスタの NameNode は同じ構成である．各クラスタの詳細な構成は第 4 章で述べる**表 1** と同じである．

高性能 Hadoop クラスタと低電力 Hadoop クラスタのそれぞれにおいて，Mahout を実行した．**図 1** に実行時間を示す．各フェーズにおける Mahout の実行を Hadoop クラスタごとに示している．Analysis Phase の実行時間を比較すると，低電力 Hadoop クラスタにおける実行時間は高性能 Hadoop クラスタの 3.6 倍であることが分かる．これは，Analysis Phase の実行時間は CPU のコア数やメモリサイズに依存するためである．一方，Transfer Phase は Analysis Phase と異なり，I/O 処理が支配的であり CPU コア数や大容量なメモリを必要としない．したがって，Transfer Phase の性能は高性能 Hadoop クラスタおよび低電力 Hadoop クラスタともに同程度である．

図 2 に各フェーズにおける DataNode1 台あたりの消費電力を示す．Transfer Phase において，HPS の消費電力

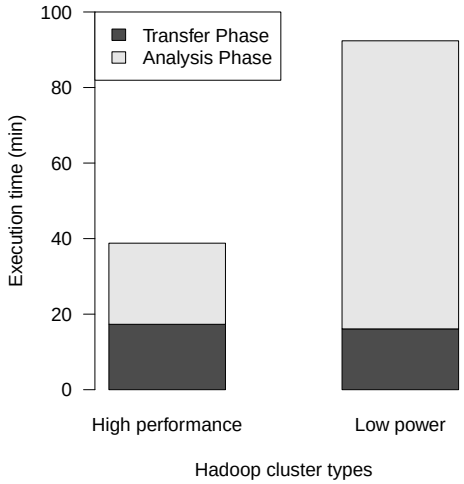

図 1 Mahout の実行時間

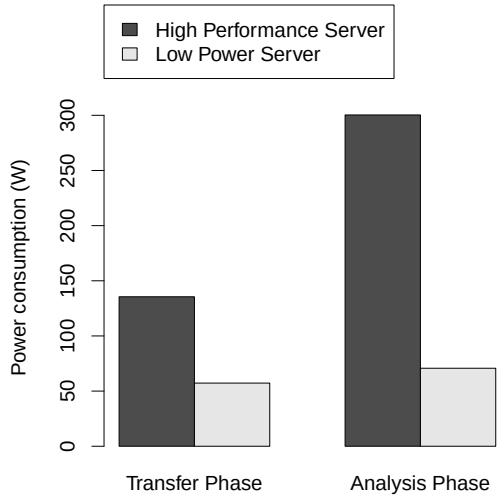

図 2 Mahout 実行時のサーバの消費電力

は LPS よりも 2.3 倍高く，Analysis Phase では 4.2 倍高い．図 1 より，Mahout を高速に実行するためには高性能 Hadoop クラスタで実行する必要があることが分かる．しかしながら，Transfer Phase の実行時間は高性能 Hadoop クラスタと低電力 Hadoop クラスタで同程度であるにも関わらず，消費電力は 2.3 倍高い．Mahout の性能を低下を抑制しつつ消費電力を削減するためには，Transfer Phase の消費電力を削減する必要がある．

2.3 Hadoop の電力効率改善

Mahout を高速かつ低電力に実行するために，フェーズごとに適した Hadoop クラスタで実行することを検討する．つまり，CPU 性能やメモリ容量を必要としない Transfer Phase は低電力 Hadoop クラスタで実行し，Analysis Phase は高性能 Hadoop クラスタで実行する．Transfer Phase の実行時間は高性能 Hadoo クラスタでも低電力 Hadoop クラスタでも同等であることから，Mahout の性能を維持しつつ Transfer Phase の電力効率を改善可能である．

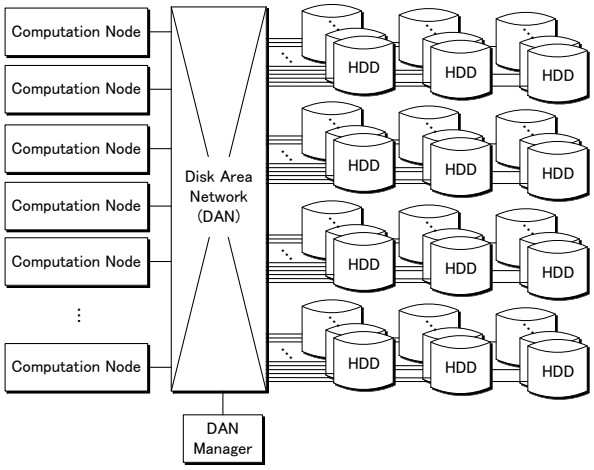

図 3 DAN を用いたアーキテクチャ例

フェーズごとに適した Hadoop クラスタで実行するためには，HDFS 上のデータ移動問題が生じる．一般に，Hadoop は Storage Area Network（SAN）による共有ストレージを用いず，Direct Attached Storage（DAS）を利用する．つまり，Hadoop クラスタを構成するサーバに内蔵されている HDD にデータを格納する．低電力 Hadoop クラスタで Transfer Phase を実行後に高性能 Hadoop クラスで Analysis Phase を実行するためには，高性能 Hadoop クラスタの HPS は低電力 Hadoop クラスタの LPS のローカル HDD に格納されたデータに高速にアクセスしなければならない．

この問題を解決するため，サーバ間で SAN による共有ストレージを使うことも考えられる．Fiber Channel（FC）を用いることで高性能な共有ストレージを構築可能であるが，FC 専用のハードウェアが必要であることから，DAS アーキテクチャと比較してコストが高い．iSCSI [8] は特別なハードウェアを用いることなく Just Bunch of Disks（JBOD）など利用して共有ストレージを実現可能である．しかしながら，Ethernet のレイテンシやネットワークトラフィックの衝突などによりディスク I/O 性能が低下する．SAN による共有ストレージは DAS を指向するアプリケーションに適しているとは言い難い．I/O 性能を DAS と同程度に保ちつつ，異なるサーバからアクセス可能なアーキテクチャが必要である．

3. Hadoop クラスタの構成変更による低電力化

3.1 Disk Area Network

フェーズごとに適した Hadoop クラスタで実行するためには，データ移動問題を解決する必要がある．そこで本稿では，この問題を解決するために著者らが提案している Disk Area Network（DAN）[13] を用いる．**図 3** に DAN を用いたアーキテクチャの例を示す．DAN スイッチはサー

バとディスク間の接続・切断の機能を提供する．DAN は SAS/SATA のプロトコルをサポートしているため，DAN スイッチ経由で接続された HDD はサーバからは DAS と同等に扱うことが可能である．ホットプラグ機能をサポートしている場合，サーバは稼働中であっても DAN スイッチを使ってディスクの接続または切断を行うことができる．サーバとディスクの接続や切断は DAN マネージャが行い，どのサーバとディスクが接続状態にあるかを管理する．

著者らは DAN スイッチとと HDD を高密度に格納可能なディスクボックスを試作し，動作の検証を行った．また，DAN を介してディスクにアクセスする際の性能を評価した結果，DAS と同等であることを確認している．1 つの DAN スイッチに 12 台のサーバと 64 台の HDD を接続できる．DAN スイッチはカスケード用のポートを備えており，最大で 10 台のスイッチを接続可能である．したがって，最大 120 台のサーバと 640 台の HDD を接続可能である．

DAN のスイッチ機能はコモディティ製品を利用して実現した．したがって，DAN スイッチ用のチップを設計する必要はないことから，DAN スイッチは低コストで製作可能である．DAN スイッチを用いることから，DAS アーキテクチャと比較して CAPEX は増加することになるが，OPEX を抑制することができれば結果として TCO の削減を実現することができる．

DAN スイッチの機能を用いることで，LPS に接続した HDD を HPS に付け替えることが可能になる．つまり，低電力 Hadoop クラスタ上で Transfe Phase を実行後，HDD の付け替えを DAN スイッチによって実現することで高性能 Hadoop クラスタにデータを移動し，Analysis Phase を実行することが可能である．

3.2 DAN を用いた Hadoop クラスタの動的構成変更

図 4 に DAN を用いた低電力 Hadoop クラスタと高性能 Hadoop クラスタの構成を示す．各サーバには DAN を介して HDD を接続する．サーバと HDD の接続状態の管理は DAN マネージャによって行われる．すべてのサーバと DAN マネージャは Ethernet によって接続される．以下の手順に従い，低電力 Hadoop クラスタから高性能 Hadoop クラスタへ HDD を付け替えて，Transfer Phase と Analysis Phase を実行する．

(1) DAN マネージャは低電力 Hadoop クラスタを構成する LPS に DAN スイッチ経由で HDD を接続．
(2) NameNode は低電力 Hadoop クラスタに HDFS を構築し Transfer Phase を実行．
(3) Transfer Phase の実行完了後，NameNode は HDFS を終了，DAN マネージャは LPS から HDD を切断し，LPS を低電力モードに切替．
(4) DAN マネージャは高性能 Hadoop クラスタを構成する HPS に HDD を接続，HPS 向け Hadoop の設定を用いて HDFS を起動．
(5) Analysis Phase を実行．

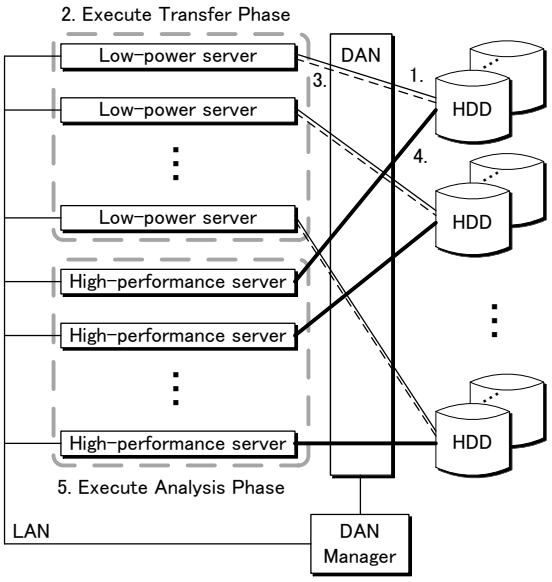

図 4 DAN を用いた Mahout の実行

LPS から HDD の接続を解除する前に，HDFS を停止させる必要がある．DAN を使って HDD を HPS に接続した後，HDFS を起動する．このとき，LPS に割り当てられていたホスト名を HPS に割り当て，HDD のマウントポイントを同じにすることで設定ファイルを変更することなく起動することが可能になる．

Transfer Phase を実行中の HPS や，Analysis Phase を実行中の HPS は他のアプリケーションの処理を実行したり，シャットダウンや低電力モードを適用して消費電力を削減することが可能である．Mahout の性能低下を抑制するためには，Transfer Phase の実行完了後すぐに Analysis Phase を実行可能な状態にあることが望ましい．たとえば，HPS をシャットダウンして電力を削減する場合を考える．このとき，Transfer Phase の実行が完全に終了した時刻から HPS を起動すると，起動処理が完了するまでの数分間は Analysis Phase を実行できないことから性能が低下する．これを回避するためには，Transfer Phase の終了時刻よりも数分前から HPS の起動処理を開始しなければならない．したがって，Transfer Phase の実行時間を見積る必要がある．Transfer Phase の実行時間は対象とするデータ量に依存することから，Transer Phase の実行前にデータ量を調べることで予測できると考えられる．なお，LPS から HPS への HDD の接続切替に要する時間は数秒程度であり [16]，Mahout の実行時間には大きな影響を与えない．

4. 評価

4.1 実験環境と評価方法

Mahout のプログラムおよび入力データは CloudSuite

表 1　Hadoop クラスタのサーバ構成

NameNode (Conventional and Proposed approaches)	
CPU	Intel Xeon E5-2690 2.9GHz 2 sockets, 8 cores/socket
Memory	DDR3 1600 LV-RDIMM 64GB
System disk	SAS 146GB 15krpm
Data disk	SATA 500GB 7.2krpm HDD x3
DataNode (HPS)	
CPU	Intel Xeon E-2690 2.9GHz 2 sockets, 8 cores/socket
Memory	DDR3 1600 LV-RDIMM 96GB
System disk	SAS 146GB 15krpm
Data disk	SATA 500GB 7.2krpm HDD x2
Host bus adaptor	SAS control card (an interface card for DAN)
DataNode (LPS)	
CPU	Intel Xeon E3-1220L 2.3GHz 1 socket, 4 cores/socket
Memory	DDR3 1600 UDIMM 32GB
System disk	SAS 146GB 15krpm
Data disk	SATA 500GB 7.2krpm HDD x2 Not included, connected via DAN
Host bus adaptor	SAS control card (an interface card for DAN)

1.0 [7] における Data Analytics を用いる．Data Analytics プログラムは Wikipedia の記事を対象に分類器を生成する．分類器を生成するために約 10GB のトレーニング用のデータと，生成した分類器をテストするために約 40GB のテスト用データが用意されている．

表 1 にサーバ構成を示す．高性能 Hadoop クラスタおよび低電力 Hadoop クラスタを構成するサーバはすべて 1Gbps の Ethernet で接続されている．提案手法では LPS と HPS の両方が必要である．本評価において，NamdeNode は 1 台，DataNode 用に LPS（Transfer Phase 用）と HPS（Analysis Phase 用）とそれぞれ同じ台数用いる．LPS と HPS には 1 台あたり 2HDD を DAN を使って割り当てる．DAN スイッチを操作するために，DAN マネージャが必要となる．本評価では，Mahout の実行フェーズと協調してサーバと HDD の接続や切断を行うことから，DAN マネージャ機能を Hadoop クラスタの NameNode 上に実装する．NameNode 上で DAN マネージャの処理が実行されることになるが，ディスクの接続・切断時以外には動作しない．したがって，ベンチマークプログラムの実行時間には影響を与えない．

従来手法として NameNode1 台，DataNode 用に 3 台の HPS を用いる．また，各 HPS は Data Disk 用の HDD2 台を内蔵し，DAN は使用しない．Transfer Phase と Analysis Phase の両方を高性能 Hadoop クラスタで実行する．

Hadoop クラスタを構成するサーバ上では，Mahout と同時に他のアプリケーション・プログラムは実行されていないものとする．Transfer Phase 実行時は HPS を停止していると仮定し消費電力は 0 とする．また，Analysis Phase 実行時，LPS はシャットダウンされていると仮定し，消費電力は 0 として評価する．提案手法における Transfer Phase の電力 P_T は式（1），Analysis Phase の電力 P_A は式（2）によって求められる．

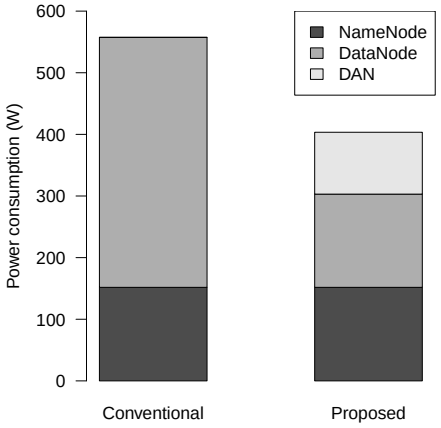

図 5　Transfer Phase の消費電力

$$P_T = P_{NN\_T} N_{NN} + P_{DN\_T} N_{DN} + P_{DAN\_T} + P_{HDD} * N_{DN} \quad (1)$$

$$P_A = P_{NN\_A} N_{NN} + P_{DN\_A} N_{DN} + P_{DAN\_A} + P_{HDD} * N_{DN} \quad (2)$$

ここで，P_{NN_T}，P_{DN_T}，P_{DAN_T} は Transer Phase の，P_{NN_A}，P_{DN_A}，P_{DAN_A} は Analysis Phase における NameNode，DataNode，DAN の 1 台あたりの消費電力である．N_{NN}，N_{DN} はそれぞれ NameNode，DataNode の数である．第 3 章で述べたとおり，Mahout の性能低下を抑制するために，HPS は Transfe Phase を実行する前に起動する必要がある．本評価では切替に伴う性能および電力オーバーヘッドは含まず，提案手法による最大の効果を調査する．

4.2　消費電力削減効果

図 5 に Transfer Phase 実行時の総消費電を示す．縦軸は消費電力であり，NameNode，DataNode および DAN それぞれの消費電力を示している．従来手法では DataNode の消費電力が 405.4W であるのに対し，LPS を用いる提案手法では 151.3W に削減している．従来手法と提案手法では同じ NameNode サーバを利用することから，提案手法によって NamdeNode の消費電力を削減することはできない．したがって，NameNode の消費電力はほぼ同じ約 150.1W である．提案手法では LPS と HPS とで切り替える必要があるため DAN を使用しなければならない．著者らが試作した DAN スイッチの消費電力は 73.3W であり，HDD を含めた DAN の消費電力は 100.4W である．DAN の増加分を考慮しても，提案手法は従来手法よりも 28%低電力であることが分かった．

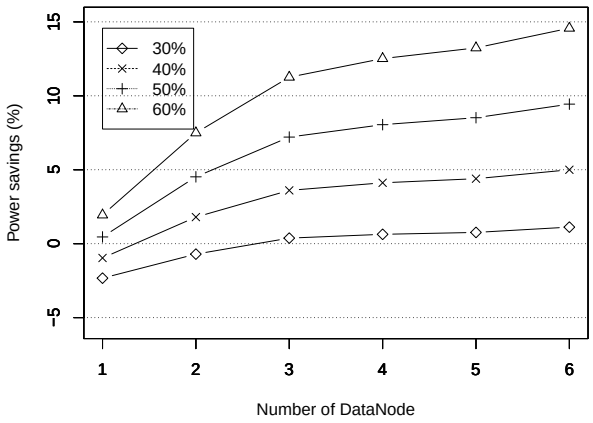

図 6　DataNode 数と消費電力削減効果

4.3　実行時間のオーバーヘッド

Transfer Phase と Analysis Phase の実行時間を合わせた Mahout の全実行時間は，従来手法が 38.7 分であったのに対し，提案手法は 40.3 分であり，4.3%増加した．Transfer Phase の実行時間はほぼ同じであった．しかしながら，Analysis Phase の実行時間が従来手法では 20.9 分，提案手法が 22.5 分であり 7.9%増加している．提案手法は LPS から HPS へと切り替えて実行することからメモリ上にデータがないことが原因であると考えられる．

この問題を解決するには，メモリを HDD と同じように LPS と HPS とで切り替え可能なアーキテクチャを導入することが考えられる．また，LPS のメモリを Remote Direct Memory Access（RDMA）によって HPS から参照できるようにする手法も考えられる．実行時間のオーバーヘッド低減技術の詳細については今後の課題である．

4.4　入力データサイズと DataNode 数の影響

提案手法はプログラム実行時間における Transfer Phase の実行時間と Analysis Phase の実行時間の占める割合によって効果が異なる．提案手法により消費電力を削減できるのは Transfer Phase のみである．したがって，Transfer Phase が占める割合が高い程，提案手法における消費電力の削減効果は大きい．CloudSuite の Data Analytics ベンチマークに用意されている入力データを用いた場合，Transfer Phase が占める割合は約 45%である．さらに，Transfer Phase の実行時間が Mahout の全実行時間に占める割合は，DataNode の数にも依存している．並列実行により Analytics Phase の実行時間が短縮されるためである．

Transfer Phase が実行時間に占める割合と DataNode 数による電力の削減効果の関係を，入力データサイズと DataNode 数との関係に基づき試算した．その結果を**図 6** に示す．縦軸は従来手法と比較して提案手法によって削減される電力の割合であり，横軸は DataNode の数である．Mahout の全実行時間に対して，Transfer Phase の実行時間が占める割合が 30%，40%，50%，60%の場合における提案手法を効果を示している．Transfer Phase の実行時間割合が 30%の場合，DataNode が 3 台以上でに提案手法の効果が確認できる．割合が 60%における DataNode 数が 6 の場合では，消費電力を約 14.6%削減できる．割合と DataNode 数が増加すると提案手法の効果が大きくなることが分かる．

5.　関連研究

これまで，データセンタの消費電力削減を目的とした研究は多く進められている．サーバのコンポーネントを対象とした研究として，CPU に対する DVFS がある．特にサーバにおいて CPU の消費電力は支配的であったことから，CPU を対象とした研究は多い．CPU の負荷に応じて消費電力が増加するようになっており，エネルギー・プロポーショナリティの改善が進んでいると言える．一方，メモリは負荷と電力が比例しない傾向にあり，エネルギー・プロポーショナリティの低さが指摘されている [3]．この問題を解決するため，メモリに対しても DFS を適用する手法も提案されている [6]．

サーバアーキテクチャレベルで電力効率を改善する手法も提案されている．KnightShift [14] は単一ノード内に高性能プロセッサと低電力なプロセッサを搭載し，負荷状況に応じて使い分ける．電力のドメインをプロセッサごとに分割することによって，エネルギー・プロポーショナリティを改善している．アイドル時の電力を削減する PowerNap という手法も提案されている [11]．時間的に細粒度にサーバの状態をスリープにすることでアイドル時の電力を削減している．

本稿における提案手法もこのサーバアーキテクチャレベルに分類される．提案手法はサーバ自体のアーキテクチャは変更する必要がないことから，既存のサーバを流用可能であるという利点がある．さらに，サーバノード内に特性の異なるプロセッサを混載する必要はないため，CPU の世代交代が容易である．また，PowerNap [11] などの電源管理技術を，著者らが提案する手法と組み合わせて用いることも可能である．

Hadoop の電力効率を改善する手法も提案されている [9], [10]．これらの手法はスケジューリングやデータの配置の工夫などを行なっているため，ハードウェアを新たに用意しないで良いというメリットがある．一方，提案手法は DAN を必要とするが，Hadoop に限らず他のアプリケーション・プログラムにおいても，Transfer Phase や Analysis Phase と同様の特徴があれば適用可能である．

DAN を用いて HDD をサーバ間で付け替えることで，他にも様々な効果を得ることが可能である．オブジェクトストレージを対象に，サーバ故障時の復旧を高速にする手法や [16]，DAN に接続して SSD の性能低下を抑制する手法 [15] が提案されている．本稿における提案手法は DAN

による HDD の切替という点でこれらの研究と同じだが，Mahout のフェーズを考慮して Hadoop クラスタの電力を削減するという点で異なる．

6. おわりに

本稿では Hadoop 上で Mahout を実行し，実行時間と消費電力を調査した．その結果，CPU 性能やメモリ容量を必要としない Transfer Phase と，高い CPU 性能と大容量のメモリを必要とする Analysis Phase があることが分かった．フェーズごとに異なるサーバ構成で実行することで，Hadoop クラスタの消費電力を削減する手法を提案した．提案手法では，サーバ間のデータ移動のために DAN を用いた．これにより，低電力なサーバで Transfer Phase を実行し，DAN スイッチを使って HDD を高性能サーバに付け替え，Analysis Phase を実行することで Transfer Phase の電力を約 28%削減可能であることを確認した．

Hadoop は処理対象のデータを HDFS へと移動させる必要がある．この操作はディスク I/O 性能が重要であり CPU 性能やメモリ容量は必要でないと考えられることから，Mahout に限らず提案手法を適用できる．アプリケーション・プログラムの実行時間のうち，Transfer Phase の実行時間が占める割合が高い程，提案手法の効果が期待される．また，アプリケーション・プログラムの実行時間が長い程，提案手法のオーバヘッドである HDD の切替時間の割合が小さくなり，提案手法の効果がより高くなる．

提案手法の適用により，Mahout の実行時間が 4.3%増加することも分かった．今後，性能オーバーヘッドを低減する手法を検討する必要がある．また，提案手法を実際にデータセンタに適用するためには，いくつかの課題が残されている．データセンタでは複数のアプリケーションプログラムが同時に実行される．他のプログラムのスケジューリングを考慮して，本手法の適用した際の効果に関して今後調査が必要である．本手法は LPS および HPS それぞれに対して DataNode の数必要となる．比較的負荷の低い Transfer Phase の実行において NameNode 数を削減するなどの対策を検討する必要がある．

参考文献

[1] Apache Hadoop: `http://hadoop.apache.org/`.
[2] Apache Mahout: `http://mahout.apache.org/`.
[3] Barroso, L. A., Clidaras, J. and Hölzle, U.: *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines, Second Edition*, Synthesis Lectures on Computer Architecture, Morgan & Claypool Publishers (2013).
[4] Dean, J. and Ghemawat, S.: MapReduce: simplified data processing on large clusters, *Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6* (2004).
[5] Delimitrou, C. and Kozyrakis, C.: Quasar: Resource-efficient and QoS-aware Cluster Management, *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 127–144 (2014).
[6] Deng, Q., Meisner, D., Ramos, L., Wenisch, T. F. and Bianchini, R.: MemScale: Active Low-power Modes for Main Memory, *Proceedings of the Sixteenth International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 225–238 (2011).
[7] Ferdman, M., Adileh, A., Kocberber, O., Volos, S., Alisafaee, M., Jevdjic, D., Kaynak, C., Popescu, A. D., Ailamaki, A. and Falsafi, B.: Clearing the clouds: a study of emerging scale-out workloads on modern hardware, *Proceedings of the Seventeenth International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 37–48 (2012).
[8] iSCSI: `http://www.ietf.org/rfc/rfc3720.txt`.
[9] Kaushik, R. T. and Bhandarkar, M.: GreenHDFS: towards an energy-conserving, storage-efficient, hybrid Hadoop compute cluster, *HotPower*, pp. 1–9 (2010).
[10] Leverich, J. and Kozyrakis, C.: On the energy (in)efficiency of Hadoop clusters, *SIGOPS Oper. Syst. Rev.*, Vol. 44, No. 1, pp. 61–65 (2010).
[11] Meisner, D., Gold, B. T. and Wenisch, T. F.: PowerNap: Eliminating Server Idle Power, *Proceedings of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 205–216 (2009).
[12] Owen, S., Anil, R., Dunning, T. and Friedman, E.: *Mahout in Action* (2010).
[13] Takashi, M., Kazuichi, O., Jun, T., Tsuyoshi, Y. and Hiroyuki, Y.: New System Architecture for Next-Generation Green Data Centers:Mangrove, *FUJITSU Scientific & Technical Journal*, Vol. 48, No. 2, pp. 184–191 (2012).
[14] Wong, D. and Annavaram, M.: KnightShift: Scaling the Energy Proportionality Wall Through Server-Level Heterogeneity, *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 119–130 (2012).
[15] 小野貴継，谷本輝夫，三吉貴史：ディスクエリアネットワークを用いたフラッシュストレージの性能改善手法，電子情報通信学会技術研究報告，Vol. 113, No. 169, pp. 7–12 (2013).
[16] 小西洋太郎，小野貴継，三吉貴史：ディスクエリアネットワークを用いたオブジェクトストレージの高速なデータ復旧手法，情報処理学会論文誌コンピューティングシステム（ACS），Vol. 6, No. 4, pp. 38–48 (2013).