大規模データアナリティクスに関する研究動向と展望

石川　佳治[†a)]

Research Trend and Future Prospects for Large-Scale Data Analytics

Yoshiharu ISHIKAWA[†a)]

あらまし　ビッグデータ時代を迎えて，膨大なデータに対し高度な分析を適用するデータアナリティクスに注目が集まっている．本論文ではデータ工学の立場から，データアナリティクスに対する研究開発の現状を分析し，今後の展望について述べる．まず，データアナリティクスに対するさまざまなアプローチを分類し，分析処理のための DBMS の拡張について説明する．また，DBMS に機械学習機能を導入する試みや，DBMS をシミュレーションエンジンとして用いる研究を紹介する．次いで，データアナリティクスの立場から，並列 DBMS と MapReduce の比較を行い，システムアーキテクチャを分類する．最後に，データアナリティクスのための MapReduce の機能拡張について触れ，今後の展望を述べる．

キーワード　データアナリティクス，データベースシステム，MapReduce，統計処理・機械学習

1. まえがき

ビッグデータが今日の情報社会で大きな話題の一つとなっているが，これに伴い**データアナリティクス**（data analytics）が重要なキーワードとして用いられるようになってきた．データアナリティクスに対する明確な定義はないが，おおまかには大規模なデータに対して高度な分析を行うことを指している．蓄積されたデータの分析に関しては，従来よりデータウェアハウスの研究・開発が進められ，既に広く活用されている．データアナリティクスとデータウェアハウスは，データの分析という点で目的や技術を共有しているが，データアナリティクスと言った場合，以下の 2 点がしばしば意図されている．

（1）　機械学習や高度な統計分析技術の活用：データウェアハウスでは，集計処理を中心とした比較的簡単な統計処理を行っていたが，データアナリティクスではより高度かつ複雑な分析を行う．

（2）　スケーラブルな処理：クラウドコンピューティングなど，今日のシステム技術を効果的に組み合わせ，大規模なデータの処理にも対応する．

† 名古屋大学大学院情報科学研究科，名古屋市
Graduate School of Information Science, Nagoya University, Nagoya-shi, 464–8601 Japan
a) E-mail: ishikawa@is.nagoya-u.ac.jp

データウェアハウスと共通するのは，対話的な分析を支援するために相応の応答時間が求められるということである．高度な分析には試行錯誤なども含む非定型の処理がより含まれるためである．

システム的な観点では，データアナリティクスには Hadoop(注1) を用いよとしばしば述べられるが，必ずしもそれは正しくない．Hadoop を用いた場合，処理記述が低レベルになってしまうことや，応答時間が大きいといった問題が存在する．このような背景をもとに，データ工学の研究分野では，データアナリティクスを支援するためのさまざまな研究開発が進められている．本論文ではこのような研究動向を概説し，今後の展望について述べる．

以降の構成は次のようになる．まず **2.** では，データアナリティクスのためのシステムのアプローチを分類する．**3.** では，DBMS を拡張し，ユーザに対し高度な分析機能を与えるアプローチについて概説する．**5.** では，分析の考え方を更に推し進め，DBMS をシミュレーションエンジンとして用いる研究について紹介する．**4.** では，DBMS に機械学習や高度な統計処理の機能を導入するための研究動向について述べる．**6.** では実世界のデータアナリティクスシステムに話題を転じ，システム構築の基盤技術について概説する．

(注1)：http://hadoop.apache.org/

7. では，MapReduce 技術に焦点を当て，データアナリティクスに関する機能拡張について紹介する．最後に **8.** で本論文のまとめを行い，今後の展望について述べる．

2. アプローチの分類

データアナリティクスに関してさまざまなシステムが提案されているが，分析者に対してどのような機能・インタフェースを提供するかという観点から分類することができる．

（1） **DBMS の拡張**：従来の DBMS を拡張し，高度な分析機能を入れていこうというものである．データウェアハウスからの自然な展開と位置づけることもできる．SQL をベースとした高レベルな分析が行えることから，データベースシステムの利用に慣れたユーザには学習の手間が小さいが，細かい処理の記述は難しいという問題もある．これについては，**3.** から **5.** でその詳細について述べる．

（2） **MapReduce ベースのアプローチ**：既に広く利用されている MapReduce [1] のアプローチに基づき，典型的にはそのオープンソース版である Apache Hadoop のフレームワークを利用するものである．インタフェースはそのままでシステムレベルの効率化を行っているものや，データアナリティクスに求められる機能を追加しているものなどがある．他のアプローチに比べ，低レベルのプログラミングが必要となる．このアプローチについても，詳しくは後述する．

（3） **統計分析言語の拡張**：R や Matlab のように，既に統計分析などで定着している言語をベースとし，それらを並列分散処理できるデータ処理基盤を構築するアプローチである．R 言語の並列実行環境は既に存在しているが，分析者が並列化を意識してプログラミングする必要があるため負担が大きい [2]．

一方，IBM の System ML は大規模な機械学習の支援を想定しており，R 言語に似たプログラミング言語で書かれたプログラムを MapReduce プログラムに変換して実行する [3]．相対的にユーザの負担は小さい．Oracle R Enterprise(注2) もこの種のシステムの一つであり，R プログラムと Oracle 及び Hadoop の処理が統合されている．Ricardo は R と Hadoop を連携するシステムであり，Hadoop に対する問合せ言語 Jaql を用いて実装されている [4]．

（4） **フレームワークベースのアプローチ**：個々の統計処理・機械学習アルゴリズムの集合と，それに伴うアルゴリズム記述のためのマクロ的なプログラミング機能を与える．比較的低レベルの処理となり，典型的には，分析処理における共通の処理を自動化するためのテンプレートが提供される．

Apache Mahout(注3) は，Hadoop 上での機械学習機能を提供している．機械学習アルゴリズムのライブラリと，このライブラリを拡張するためのテンプレートが提供される．Google の Pregel [5] はバルク同期並列（bulk synchronization parallel, BSP）モデルに基づくグラフ処理の並列実行フレームワークであり，その考え方をもとにオープンソース化したものが Giraph(注4) である．GraphLab(注5) も，大規模グラフに対する機械学習などの処理を高レベルのインタフェースを用いて記述するフレームワークである．マルチコアシステムの並列性を活用するための抽象化を行い，graph-parallel な実行を実現する．グラフマイニングをターゲットとしたフレームワークとしては，Pegasus(注6) もある [6]．Hadoop 上における行列とベクトルの乗算の一般化した繰返しを効率良く実装している．

Preferred Infrastructure 社及び NTT ソフトウェアイノベーションセンタにより開発された大規模データ分析基盤 Jubatus(注7) は，実時間のストリーム処理に焦点を当て，機械学習の機能を提供しているシステムである [7]．MapReduce がバッチ処理であるのに対し，オンライン処理を対象としている．データ分析は Update（モデルの更新），Analyze（現在のモデルによるデータの分析），Mix（二つのモデルをマージする）という三つの操作に基づいて定義され，ユーザは Update と Analyze についてその内容を記述する．

3. 分析インタフェースとしての DBMS

DBMS にデータアナリティクスの機能を導入することは，データベースに慣れ親しんでいるユーザにとっては，学習コストが少なくて済むというメリットがある．SQL による宣言的なデータ操作に基づく高レベル

(注2)：http://www.oracle.com/technetwork/database/options/advanced-analytics/r-enterprise

(注3)：http://mahout.apache.org/
(注4)：http://giraph.apache.org/
(注5)：http://graphlab.org/
(注6)：http://www.cs.cmu.edu/~pegasus/
(注7)：http://jubat.us/

の分析が比較的容易に記述でき，対話的な分析に適しているといえる．効率の面については，データベースに格納されているデータを外部に抽出せずに分析処理が行えるため，オーバヘッドが小さいという利点もある．以下では注目すべき研究事例について紹介する．

3.1 MADlib：統計処理・機械学習機能の導入

UCB と EMC Greenplum グループなどにより共同開発されている MADlib(注8)では，DBMS に統計処理・機械学習の機能を導入しており，SQL を用いてデータ分析タスクを記述することができる [8], [9]．DBMS 内に存在するデータに対し統計処理や機械学習の処理を直接的に適用することができ，実行時間の短縮を実現している．

現在のシステムでは，実装例として最小 2 乗法，ロジスティック回帰，SVM に基づく分類，クラスタリング（k-means 法），サンプリングなどが実現されている．特定の統計処理・機械学習の機能を実現・利用するには，SQL と Python 及び C++ 言語によるユーザ定義関数を用いる．統計処理や機械学習では，しばしば線形代数の処理が必要となることがある．また，機械学習の過程には最適化処理が含まれることが多く，そのためには収束までの繰返し処理が必要となる．MADlib では，今日の DBMS 技術を効果的に活かしてこれらの機能を実現している．

簡単な例として，`x`, `y` という数値属性をもつリレーション `data` に対し線形回帰により直線のあてはめを行う場合には，

```
SELECT (linregr(y, x)).* FROM data
```

というユーザ定義関数の呼び出しを含む問合せとなる．ユーザ定義関数内で `data` テーブルへのアクセスが行われる．

このような実装のアプローチには欠点も存在する．DBMS 内の拡張部分が SQL，ユーザ定義関数，及びドライバプログラムの組合せになってしまい，管理が難しくなる．また，並列化が必ずしも容易ではないことも問題点である．

3.2 SciDB：大規模配列の直接的な支援

SciDB(注9)は，Stonebraker らにより，科学分野におけるデータアナリティクスを対象として構築されたデータベースシステムである [10]．故 Jim Gray は，大規模なデータの処理に基づく研究の形態が科学の新たな第 4 のパラダイムであると述べた [11]．科学分野のデータ処理においては配列を使った処理が多く出現することから，大規模配列を効率的かつ容易に操作できる DBMS を構築しようというのが開発の動機である．高レベルの抽象化を行うことにより，科学者による分析を容易にしている．

例えば，

```
CREATE ARRAY Sensor_Data <WindSpeed: double,
   Temperature: double, Conditions: string>
   [SensorID(string), Timestep]
```

という定義により，`SensorID` と `Timestep` という 2 次元からなる配列が定義される．後者は 1 から 1 ステップずつ増加する時刻に対応する．配列の各要素は三つの属性からなる．2 次元配列であるので，例えばセンサ ID が `A` であるセンサの時刻 5 のときの値は `Sensor_Data["A", 5] = (12.5, 75.1, clear)` といった具合になる．SciDB では AQL (Array Query Language) という問合せ言語が提案されており，配列へのアクセスを含む

```
SELECT S.Temperature
FROM Sensor_Data S
WHERE S.WindSpeed < 10 AND
   S.Timestep BETWEEN 550 and 650
```

といった問合せが記述できる．また，大規模配列に対する統計処理や線形代数による操作機能なども提供されている．

システム的な特徴としては，大規模疎行列を効率的に格納・操作するための技術や，並列処理技術が導入されている．また，R 言語との連携もなされており，R プログラムから SciDB へ容易にアクセスすることができる．

3.3 モデルベースのビュー

統計処理や機械学習の結果として得られた情報をユーザにどのように提示するかも重要なポイントである．そのような情報を表現できるようなデータモデルを構築し，DBMS と統合するという考え方がある．従来の RDBMS の上位に抽象化のレイヤを設けるイメージであり，統一された抽象化で統計的処理の詳細をユーザから隠ぺいする．データベースとして抽象化することで，分析結果が把握しやすくなり，問合せを用いたデータ操作も可能となる．

この考え方を初期に打ち出したものとして，MauveDB [12] がある．そこでは，統計モデルを RDBMS と統合するために，ビューの概念を拡張した

(注8)：http://madlib.net/
(注9)：http://scidb.org/

モデルベースのビュー（model-based view）の概念を提案している．以下は，さまざまな地点における気温の測定値が `raw-readings` というテーブルに格納されているとき，回帰分析に基づくモデルベースのビューを構築する例である（少し簡略化している）．

```
CREATE VIEW RegView(x[0:9:.1], y[0:9:.1], temp)
   AS FIT temp USING x, y
   BASES 1, x, x^2, y, y^2
   TRAINING_DATA SELECT x, y
   FROM raw-readings
```

生データの変数 x, y を説明変数として，x, x^2, y, y^2 の項を用いて回帰分析を行うことが指定されている．構築した `RegView` に対しては，x-y 平面の任意の点(注10)に対し，たとえ測定値がなくとも問合せが行える．システムは回帰分析のあてはめにより推定された値を返す．このような考え方は，関数近似によるビューを提供する FunctionDB [13] でも用いられている．また [14] では，データベース中のデータを SVM により自動分類し，分類属性（例：論文データベースにおける `category` 属性）を自動付与する分類ビュー（classification view）が実現されている．

センサデータベースシステムにおける問合せ処理に関する論文 [15] では，無線センサネットワーク上にモデルベースのビューを構築するアプローチをとっている．センサから取得されるデータに対しガウス分布に基づく確率モデルをあてはめ，問合せ処理において確率的推論を実施する．これにより，センサに対する問合せに対して，指定された精度の条件を満たすのであれば必ずしもセンシングは実施せず，モデルに基づく推定結果を返すことで対応する．

近年，確率的情報を直接的に表現し操作可能とする**確率的データベース**（probabilistic database）に関する研究が進んでいる [16], [17]．確率的データベースは，統計処理や機械学習により得られる確率的情報に対するインタフェースとしても機能することから，上記のような流れと親和性が高い．例えば，BayesStore [18] は，DBMS の内部にベイジアンネットワークに基づく確率モデルを保持することができ，問合せ処理においては欠損したデータに対して確率的な推論に基づく処理を実現できる．

(注10)：実際にはシステムの実装上，離散化がされている．この例では，x, y 座標がそれぞれ 0 以上 9 以下の範囲について，0.1 きざみのグリッドが構築される．

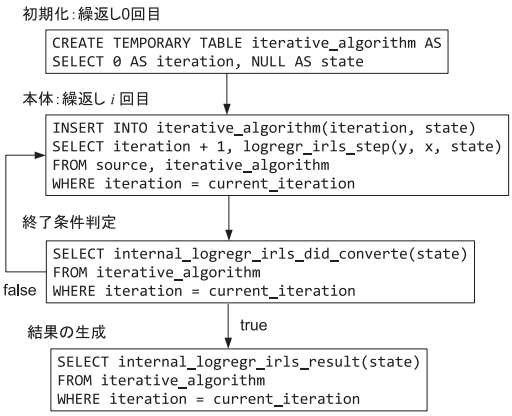

図 1　繰返し処理による最適化
Fig. 1　Optimization based on iterations.

4. DBMS への学習機能の導入

DBMS に統計処理・機械学習の機能を導入するためのさまざまなアプローチについて説明する．

4.1 最適化・繰返し処理の実現

機械学習のアルゴリズムにはしばしば最適化処理が含まれており，そこでは繰返し処理が行われる．通常の DBMS は繰返し処理に対しては十分な支援がないため，何らかの工夫によりそのような機能を実現することになる．

MADlib [8], [9] では，RDBMS 内の n 回の繰返しを，n 行からなる仮想テーブルを作成することで実現している．実際には，C++ 等による組込みのドライバルーチン内で複数の SQL 問合せを呼び出すことで対処する．繰返しによりロジスティック回帰を行う例を図 1 に示す．ここでは四つの SQL 問合せをドライバプログラムから呼び出している．なお，`current_iteration` はプログラム変数であり，初期値は 0 で繰返しのたびにインクリメントされる．

この考え方の発展として，Bismarck [19] では凸計画法（convex programming）の DBMS 内での効率的な実現を行っている．凸計画法は，SVM，最小 2 乗法，ロジスティック回帰，条件付き確率場（CRF）などの統計処理・機械学習の実装で用いられる．凸計画法の機能を用いることで，統計・学習処理の実装が容易に行えるという利点がある．

4.2 個別の学習機能の導入

統計処理・機械学習の個々の手法に着目して DBMS への実装技術を開発した研究もある．Wang らはデータベース中に存在するテキストからの情

報抽出とそれを用いた問合せ処理を実現している [20]．例えば，住所録データベースのあるタプルに "2181 Shattuck North Berkeley CA USA" という文字列が格納されているとき，条件付き確率場（CRF）に基づき事前に学習されたモデルにより，これを street_num，street_name，city，state，country という五つの属性からなるモデルベースのビューの各属性に対応づける．これにより，SQL 問合せにおいて，実際には存在しない属性 city に対し WHERE city = 'Berkeley' という条件指定を可能とする．実際には，情報抽出処理の一部は，問合せ時に動的に Viterbi アルゴリズムを用いて行われ，与えられた条件にマッチする可能性の高い文字列が選ばれる．同じ著者は，このアプローチを更に拡張し，マルコフ連鎖モンテカルロ法（MCMC）に基づく推論も DBMS に導入している [21]．

そのほか，グラフィカルモデルと MCMC に基づく確率的推論を DBMS で実行する研究は [22] にも見られる．CRF に基づく情報抽出による確率的データベースの構築手法に関する議論は [23] にある．

5. シミュレーションエンジンとしての DBMS

これまで述べたアプローチは，主として既存のデータの分析の効率化のために DBMS を用いようというものであった．一方，より斬新な考え方として，DBMS をシミュレーションエンジンとして用いて未来を予測しようとするアプローチがある．シミュレーションの実行をデータベースに対する問合せのような感覚で指定することができる．まだ事例は多くはないが興味深い研究トピックである．

代表例として，Rice 大学や IBM により開発された MCDB（Monte Carlo Database）[24] について説明する．MCDB もモデルベースのビューの考え方を用いる．ここでは販売部門の売り上げを予測する例を示す．モデルベースのビューを次のように定義する．ただし，顧客の企業の ID とその所在地域に関する情報が実リレーション CUST_ATTRS(ID, AREA) に収められているとする．

```
CREATE TABLE SALES(CID, AMOUNT) AS
FOR EACH d in CUST_ATTRS WITH MONEY AS Gamma(
   (SELECT n.SHAPE FROM AMT_SHAPE n
    WHERE n.CID = d.CID),
   (SELECT sc.SCALE FROM AMT_SCALE sc
    WHERE sc.REGION = d.REGION))
SELECT d.CID, m.VALUE FROM MONEY m
```

ここでは各顧客に対する売上がガンマ分布でモデル化でき，ガンマ分布の形状（shape）パラメータは顧客に依存し，尺度（scale）パラメータは顧客の所在地域に依存していると想定している（ガンマ分布によるモデル化の詳細は省略）．内側の二つの問合せでは，パラメータ表 AMT_SHAPE, AMT_SCALE からガンマ分布の該当するパラメータ値を取得している．パラメータ値をガンマ関数に適用することで，各顧客について売上高の推測値（ガンマ分布に従う乱数）が 1 行 1 属性のリレーション MONEY（2 行目）に入る．最後の行で，顧客 ID と売上高の推測値をペアにしてモデルベースのビュー SALES に挿入する．

MCDB では，このように定義されたビュー SALES に対して問合せを行うことができる．シミュレーションであるため，個別の値よりは「売上の予想額を求めよ」といった集約問合せが主体となる．MCDB は，実際にはこのような問合せが与えられた時点でランダムサンプルを生成して問合せに答える．推測の精度を高めるため，サンプルは多数生成され，それらに対する平均が計算される．MCDB ではこのようなサンプリング処理を効率化するためのさまざまな工夫がなされている．上記の例では既存のガンマ関数を用いたが，ユーザ定義関数を用いることも可能である．

MCDB を発展させた形で，同じグループにより SimSQL [25] が開発されている．マルコフ連鎖モンテカルロ法（MCMC）に基づくシミュレーションを SQL 風に宣言的に記述することが可能であり，ベイズ推論に基づく機械学習プログラムをわずかな行数で表現できる．ポイントは，多段の繰返しによる推論処理をシステム側で実現するところにある．大規模な推論を効率化するための問合せ最適化技術が提案されている．

6. システム基盤技術の現状

これまでは高度な統計処理や機械学習の DBMS への導入について述べてきた．ここでは話題を転じ，データアナリティクスのシステム基盤技術に焦点を当てる．想定する処理要求は，従来のデータウェアハウスの延長上に位置する問合せ処理であり，企業等の大規模データ分析業務への対応が典型的なシナリオとなる．

6.1 並列 DBMS か MapReduce か

まず，大規模データに対する高速な処理を求める

データ分析要求があり，何らかのシステムを導入する場合に，並列 DBMS がよいのか，若しくは MapReduce（Hadoop）がよいのかという議論がある．今日の DBMS 技術は，並列処理以外にも，大容量化する主記憶への対応，フラッシュメモリ（SSD）の活用，マルチコア，GPU の活用など，進化するハードウェア技術を活かした高速化が進んでいる．また，RDB のテーブルをタプル（行）単位で管理せず，カラム（列）単位で管理するカラムストア技術も有効である．これらについては宮崎の解説が詳しい [26]．一方，MapReduce に関してもさまざまな研究開発が行われている．[27], [28] にサーベイがあるが，新しい技術であるため開発の余地が大きい．

並列 DBMS と MapReduce の比較に関しては [29] において詳細な実験が行われており，その結果に基づく解説が [30] にある．[29] の実験では，2 種類の DBMS（カラムストア型 DBMS の Vertica と商用 RDBMS）と Hadoop が対象とされ，1) 全レコードに対する簡単な文字列照合，2) Web のログのグループ化処理，3) 結合演算を含む処理について比較が行われている．いずれにおいても Hadoop が二つの DBMS に劣っており，Vertica が 1), 2) で，商用 DBMS が 3) で最もよい性能を示した．本来 Hadoop が得意とされる 1) についても性能が劣る理由については，

- Hadoop では実行時にテキスト形式のレコードを読み込み解析する必要があり，このコストが大きい．一方，DBMS は事前のデータロード時に内部フォーマットに変更しておくことができる．
- DBMS では蓄積したデータの圧縮を行うことができ，問合せ処理では有利に働く．特にカラムストアでは圧縮の効果が大きい．
- 並列 DBMS では問合せ処理の部分処理間のパイプライン化が進められているのに対し，Hadoop では各ステップごとにファイルシステムに書き出して再読み込みを行うため，オーバヘッドが大きい．
- 並列 DBMS では問合せ処理において各ノードにどのようなタスクを割り振るかを細かく制御し，効率を高めている．一方，Hadoop のタスクスケジューリングの粒度はより粗く，オーバーヘッドが大きい．

などの説明がなされている．

ただしこれには，

- そもそも Hadoop は DBMS が対象としないようなタスクにも適用されており，ベンチマークの設定が不適切である．例えば，Google の検索のために転置索引を作る処理は DBMS では行えない．
- 並列 DBMS ではデータをロードするコストが大きい．
- Hadoop の実装では高度なフォールトトレランスが実現されており，ノードに障害が発生しても実行処理を継続できる．
- 並列 DBMS では，システムの設定やパラメータの調整に高度な知識を必要とする．
- 現実に利用できる並列 DBMS は全て商用であり，導入コストが大きい．

といった反論もある．

結局のところ，並列 DBMS と Hadoop のどちらかという問いについては，状況に応じて使い分けるということになる．同じデータを何度も利用し，可能な限り高速化したいという場合には並列 DBMS が適している．Hadoop が向く応用は，ETL (extraction-transformation-load) ツールとしての read-once なデータの処理，DBMS では行いにくい複雑な分析や半構造データ（semi-structured data）の処理，すぐに大まかな分析を行いたい場合の処理などが挙げられる．

6.2 システムアーキテクチャの分類

現状のシステムは，システムアーキテクチャの面から大きく四つに分けられる．

（1） **MapReduce ベース**：MapReduce プログラムを直接書くということも選択肢であるが，対話的分析を考えると問合せ機能があることが望ましい．高レベルの問合せを MapReduce ジョブに変換することが一つのアプローチとなる．

Apache Pig プロジェクト(注11)において開発されている Pig Latin 言語は，Hadoop 上に位置するデータフロー型の問合せ言語である [31]．例えば，SQL による

```
SELECT category, AVG(pagerank)
FROM urls WHERE pagerank > 0.2
GROUP BY category HAVING COUNT(*) > 10^6
```

という問合せが，Pig Latin では

```
good_urls = FILTER urls BY pagerank > 0.2;
groups = GROUP good_urls BY category;
big_groups = FILTER groups BY COUNT(good_urls) > 10^6
output = FOREACH big_groups GENERATE
           category, AVG(good_urls.pagerank);
```

と記述できる．データフローを意識する点で SQL よりは低レベルであるが，MapReduce プログラムを書

(注11)：http://pig.apache.org/

くことに比べると格段に記述が容易である．

Apache Hive プロジェクト(注12)は，同様に Hadoop 上に高レベルの問合せ言語を実現するものである [32]．Hive では，よりリレーショナルデータベースに近く，テーブルの概念があり，SQL ライクな HiveQL が提供されている．SQL 風の記述もできるが，以下の例のように，ユーザが作成したカスタム化された MapReduce プログラムと連携することもできる．

```
FROM(
 MAP doctext USING 'python wc_mapper.py' AS (word, cnt)
 FROM docs
 CLUSTER BY word
)a
REDUCE word, cnt USING 'python wc_reduce.py';
```

この例では単語カウント処理を行っている．

他には，SQL92 をベースにデータウェアハウス処理などに関する SQL99 の機能（例：ROLLUP() など）を導入した，Google による Tenzing [33] がある．DBMS の最適化技術を導入し，効率化を図っている．一方，Jaql(注13)は，JSON (Javascript Object Notation) 形式フォーマットをもつデータに対する問合せ言語である．フラットなリレーショナルデータのみならず半構造データや XML データを扱うことができる．問合せは Hadoop 上の MapReduce 処理などに展開できる．また，SciHadoop [34] は，Hadoop 上での配列指向の問合せ処理を可能としており，科学分野への応用を支援している．

MapReduce はタスクの起動に遅延が大きいため，このアプローチでは，分析処理に求められる対話的問合せの応答時間が問題となる．

（2） **並列 DBMS ベース**：非共有型のアーキテクチャを有するシステムが一般に用いられる．商用システムでは，Oracle, SQL Server, DB2 以外に，Teradata, Vertica, Greenplum, Netezza, ParAccel など，さまざまなものがある．Google の Dremel [35] は入れ子構造を許すカラムストアであり，複数ノードを木構造に配置してスケーラブルな集約処理を行うことで，対話的分析を支援する．カラムストアにおいて対話性を向上させるための技術については，[36] にも議論がある．

並列 DBMS の別の流れとしては，入出力処理に焦点を当てたものがある．今日では，計算機の能力が向上する一方で，それにもましてデータが爆発的に増大している．そのため，超大規模データベースでは入出力は依然として大きいコスト要因である．喜連川らによる OoODE (Out-of-Order Database Engine) では，DBMS が非同期的に入出力を発行するアウトオブオーダ実行を用いて，ディスクストレージの入出力帯域をフルに活用している [37]．また，タスク分解による高多重のタスク並列実行により，マルチコアプロセッサの演算能力も活かしている．

（3） **MapReduce と DBMS のハイブリッド**：DBMS と MapReduce を組み合わせるアプローチも存在する．HadoopDB(注14)では，複数の単一ノードの DBMS（PostgreSQL）を，Hadoop を通信レイヤとして統合している [38]．問合せは MapReduce により並列化されるが，MapReduce タスク内のデータ処理は DBMS にプッシュされる．

SQL/MapReduce [39] は，実際には DBMS を用いてはいないが，DBMS 的な要素と MapReduce の実行方式を組み合わせたシステムである．MapReduce ライクな実行環境のもとで，ユーザ定義関数（user-defined function, UDF）を SQL 風の言語で記述する．ユーザの立場からは SQL ライクな問合せの実行に見えるが，実行処理は MapReduce 方式で行われる．

Osprey [40] は，MapReduce のアプローチにヒントを得て，並列 DBMS のフォールトトレランス性を高めたミドルウェアである．データ分析処理の問合せによっては，その処理が長時間かかるものもあり，障害時の再実行はできるだけ回避したいという要求がある．Osprey は，SQL を小さい副問合せに分割し，並列 DBMS の各ノードにスケジューリングする．

（4） **その他のアプローチ**：上記の分類には合致しない注目すべき研究について説明する．Spark [41] は，並列分散環境において機械学習処理を効率的に実現するためのフレームワークである．このシステムは，分散共有メモリの抽象化であり耐障害性を有する Resilient Distributed Dataset (RDD) を用いて構築されている．RDD では，粗い粒度の演算子（例：map, group-by, join）に限定してデータ集合を一括して処理する．また，各データセットに対するリネージ（linage）を追跡し，障害等でデータが失われたときは再計算する．これにより，上位に位置する Spark は，主記憶上に存在するデータセットが失われる心配をせ

(注12)：http://hive.apache.org/
(注13)：http://code.google.com/p/jaql
(注14)：http://db.cs.yale.edu/hadoopdb/hadoopdb.html

ずに利用できる．データアナリティクスにおいては，繰返し処理がしばしば現れ，用いられるデータセットの時空間的な局所性が高いことから，主記憶の有効利用はたいへん重要である．Spark を更に拡張し，SQL 問合せ処理と機械学習を含む複雑な分析の双方を支援するシステムが Shark である [42]．SQL の実行処理については，MapReduce 的な実行技術とデータベースシステム技術（例：カラムストア）を組み合わせている．

IBM の System ML [3] に似たシステムで，Hadoop 上で線形代数による分析処理を支援するシステムとして Cumulon がある [43]．MapReduce での処理パターンが行列演算とは必ずしも合致しないことを考慮し，線形代数の操作を分析し，処理を分割するアプローチをとっている．コストモデルに基づいて，論理プラン，物理プラン，配備（deployment）プランと段階を追って最適化処理を行う．

7. MapReduce の機能拡張

データアナリティクスにおいて MapReduce 技術を有効活用するためのさまざまな研究が行われている．幾つかのトピックを取り上げて事例を説明する．

7.1 結合処理の支援

結合（join）処理は，データの分析でしばしば発生する処理であり，またその処理コストも大きい．MapReduce を使用して結合処理を行う場合，一般的には効率が悪い処理となってしまう．例えば三つのデータ集合について $R \bowtie S \bowtie T$ という多重結合（multi-way join）を実行する場合，まず第 1 のジョブで R と S を結合し，その結果の U を HDFS ファイルシステムに書き出す．第 2 のジョブでは U と T を結合して，出力 V を再び HDFS に書き込み，第 3 のジョブで V のタプル集合をまとめる．このような処理は効率が悪いことから，結合処理のための機能拡張が研究されている．

Map-Reduce-Merge [44] は，map と reduce 演算に加え，merge 演算を導入している．二つの入力に対してそれぞれ map/reduce タスクを実行したとき，最後にそれぞれの reducer の出力を統合するのが merge 演算であり，merger が処理を担当する．Map-Join-Reduce [45] は，map と reduce の間に join 演算を加えるアプローチである．MapReduce が filtering-aggregation のプログラミングモデルであったのに対し，Map-Join-Reduce では filtering-join-aggregation というモデルとなる．MapReduce における実行処理では map と reduce の間に 1 対 1 の shuffling 処理が実行されるが，Map-Join-Reduce の shuffling 処理は 1 対多であり，結合のためにレコードを分配することができ，MapReduce では複数のジョブで実行する結合処理が 1 パスで実行できる．同様のアプローチは [46] でも提案されている．

なお，通常の MapReduce 上で結合処理を行う場合においても，実際には幾つかの処理方式の選択肢がある．[47] では，ログを MapReduce に処理する状況で，2 入力の結合を行う場合に対する複数の結合処理方式の比較を行っている．

7.2 効率的な繰返し処理

統計処理や機械学習では，しばしば繰返しによる最適化処理が行われる．結合処理でも問題となったが，MapReduce では，各ジョブが毎回出力をファイルシステムに書き出し，その結果を利用するジョブはあらためてデータの読出しを行う．繰返しの過程において多くのデータの値が変わらない場合でも毎回データを書き出すのは無駄が多い．また，収束条件の判定を行うためだけに余分な MapReduce ジョブを実行する必要もある．

このような問題点のため，HaLoop システムでは MapReduce 環境における効率的な繰返し処理を実現している [48]．繰返しにおいて不変なデータをキャッシングする機能や，前回の reducer の出力をキャッシュしておき，停止条件の判定を容易にするなどの工夫がなされている．また，繰返し処理のためのスケジュールの制御なども加えられている．

[49] では，インクリメンタルに大規模データを処理するための汎用アーキテクチャとして CBP (continuons bulk processing) を提案している．translate という新しい演算を加えているのが特徴である．translate は状態を入力として受け取ることができ，データに一括して処理を適用できる．すなわち，CBP では状態の永続化が可能となっており，繰返しなどの分析処理では有効にはたらく．

鬼塚らは，繰返し処理を表現可能な問合せ言語 OptIQ による問合せを，MapReduce 及び Spark 上で効率的に実行するための最適化手法を示した [50]．実体化ビューの概念を活用し，繰返し時に不変の部分と変化する部分を切り分け，変化する部分のみを評価し冗長性を省くことを実現している．

MapReduce において再帰処理を実現する研究もあ

る [51]．再帰処理は，グラフ処理や PageRank，推移的閉包の計算などで活用できる．

7.3 索引の支援

MapReduce における処理は基本的にはシーケンシャルなアクセスとなってしまい，複数回のデータアクセスを行う場合，毎回無駄なコストが発生してしまう．これに対し，DBMS の問合せ処理では索引を活用することで問合せを高速化することが基本戦略となっている．このような理由から，Hadoop++ システムでは MapReduce における索引の導入・利用手法を実現している [52]．ただし，索引を作成するには，対象データのスキーマや想定される MapReduce ジョブが分かっていないといけないという制約がある．Hadoop++ の索引は選択処理だけでなく結合処理に対しても適用でき，効率的な問合せが実現できる．

Hadoop++ には索引の構築時間が大きいという問題があった．そこで，HAIL (Hadoop Aggressive Index Library) では，HDFS へのデータロード時に索引を構築するアプローチをとっている [53]．データロード自体が重たい処理であるため，索引構築のオーバヘッドは相対的には小さいものとなる．また，Hadoop ではデフォルトでデータを三つ複製しているが，それぞれの複製を異なるキーでソートしておき，個別にクラスタリング索引を構築しておく．これにより，与えられた問合せに対してより適した索引を選ぶことを可能としている．

ここで述べた以外に，MapReduce の効率化のために，複数ジョブの実行処理の共有，カラムストア型のデータ管理，データ配置の工夫，パイプライン化によるタスク間でのデータの受渡し，DBMS 的な最適化処理などさまざまな提案がなされている [27], [28]．データアナリティクスに特化した話ではないが，性能向上という点で分析処理でも恩恵が得られることになる．

8. むすび

今後どのように技術革新が進んでいくかを予測することは容易ではないが，データアナリティクスは今後の情報社会においてますます重要となっていくと考えられ，新たな研究開発が更に進展することは期待できる．本論文では並列 DBMS と MapReduce を対立するものであるかのように一部扱ってきたが，将来的にはお互いが相手の長所を取り込む形で発展が進んでいくと考えられる．

今後考えられる方向性の一つは，ストリーム的に得られる膨大な情報に対応するための，実時間分析処理の技術開発である．Jubatus については本論文でも紹介したが，Twitter による Storm システム(注15)や，Apache S4(注16)，また，最近発表された Amazon のサービスである Kinesis(注17)などはそのような流れの中にある．データストリームの処理システムについてはデータベース系のベンダーも対応を進めており，今後更なる展開が期待できる．

個別のデータに特化した研究及びフレームワークの開発も更に今後進むと考えられる．グラフデータに関する GraphLab や Giraph などはそのような例であるが，地図データに代表される時空間データ，マルチメディアデータ，ソーシャルデータなどの対象ドメインに特化することで高度な機能を提供できることが期待できる．

また，**5.** で述べたシミュレーション分析も今後大いに期待できる領域であると考える．膨大なデータと高度なデータ分析により，未来を予測しベストの行動を選択することや，ある選択肢を選んだ場合にどのような結果が生じるかの分析（what-if 分析）などは，ビッグデータを活用する一つの方向であると考える．

更に，Hadoop や並列 DBMS など，データアナリティクスに関連するシステムの開発にも継続して注目していく必要がある．Apache は 2013 年 10 月に Hadoop の新しいバージョンである Hadoop 2 をリリースしている．新たなフレームワークとして，MapReduce の実装よりもより汎用的な YARN (Yet-Another-Resource-Negotiator)(注18)が導入され，MapReduce のモデルに従っていないアプリケーションを作成・実行することも可能となっている．今後の研究開発では，このような新機能の活用も重要なポイントとなるであろう．

謝辞 本研究の経費の一部は，内閣府最先端研究開発プロジェクト（FIRST），科研費（25280039），及び，文部科学省・次世代 IT 基盤構築のための研究開発「ビッグデータ利活用のためのシステム研究等」による．

文　献

[1] J. Dean and S. Ghemawat, “MapReduce: Simplified data processing on large clusters,” CACM, vol.51, no.1, pp.107–113, 2008.

(注15)：http://storm-project.net/
(注16)：http://incubator.apache.org/s4/
(注17)：http://aws.amazon.com/jp/kinesis/
(注18)：http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html

[2] E. McCallum and S. Weston, Parallel R, O'Reilly, 2011.
[3] A. Ghoting, R. Krishnamurthy, E. Pednault, B. Reinwald, V. Sindhwani, S. Tatikonda, Y. Tian, and S. Vaithyanathan, "SystemML: Declarative machine learning on MapReduce," ICDE, pp.231–242, 2011.
[4] S. Das, Y. Sismanis, K.S. Beyer, R. Gemulla, P.J. Haas, and J. McPherson, "Ricardo: Integrating R and Hadoop," SIGMOD, pp.987–998, 2010.
[5] G. Malewicz, M.H. Austern, A.J.C. Bik, J.C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski, "Pregel: A system for large-scale graph processing," SIGMOD, pp.135–146, 2010.
[6] U. Kang, C.E. Tsourakakis, and C. Faloutsos, "PEGASUS: A peta-scale graph mining system - implementation and observations," ICDM, pp.229–238, 2009.
[7] 岡野原大輔，"大規模データ分析基盤 Jubatus によるリアルタイム機械学習，" 人工知能学会誌，vol.28, no.1, pp.98–103, 2013.
[8] J. Cohen, B. Dolan, M. Dunlap, J.M. Hellerstein, and C. Welton, "MAD skills: New analysis practices for big data," PVLDB, vol.2, no.2, pp.1481–1492, 2009.
[9] J.M. Hellerstein, C. Ré, F. Schoppmann, D.Z. Wang, E. Fratkin, A. Gorajek, K.S. Ng, C. Welton, X. Feng, K. Li, and A. Kumar, "The MADlib analytics library or MAD skills, the SQL," PVLDB, vol.5, no.12, pp.1700–1711, 2012.
[10] M. Stonebraker, P. Brown, D. Zhang, and J. Becla, "SciDB: A database management system for applications with complex analytics," IEEE Comput. Sci. Eng., vol.15, no.3, pp.54–62, 2013.
[11] T. Hey, S. Tansley, and K. Tolle, eds, The Fourth Paradigm: Data-Intensive Scientific Discovery, Microsoft Research, 2009.
[12] A. Deshpande and S. Madden, "MauveDB: Supporting model-based user views in database systems," SIGMOD, pp.73–84, 2006.
[13] A. Thiagarajan and S. Madden, "Querying continuous functions in a database system," SIGMOD, pp.791–804, 2008.
[14] M.L. Koc and C. Ré, "Incrementally maintaining classification using an RDBMS," PVLDB, vol.4, no.5, pp.302–313, 2011.
[15] A. Deshpande, C. Guestrin, S.R. Madden, J.M. Hellerstein, and W. Hong, "Model-based approximate querying in sensor networks," VLDB Journal, vol.14, no.4, pp.417–443, 2005.
[16] N. Dalvi, C. Ré, and D. Suciu, "Probabilistic databases: Diamonds in the dirt," CACM, vol.52, no.7, pp.86–94, 2009.
[17] D. Suciu, D. Olteanu, C. Ré, and C. Koch, "Probabilistic Databases," in Synthesis Lectures on Data Management, Morgan & Claypool, 2011.
[18] D.Z. Wang, E. Michelakis, M. Garofalakis, and J.M. Hellerstein, "BayesStore: Managing large, uncertain data repositories with probabilistic graphical models," PVLDB, vol.1, no.1, pp.340–351, 2008.
[19] X. Feng, A. Kumar, B. Recht, and C. Ré, "Towards a unified architecture for in-RDBMS analytics," SIGMOD, pp.325–336, 2012.
[20] D.Z. Wang, M.J. Franklin, M. Garofalakis, and J.M. Hellerstein, "Querying probabilistic information extraction," PVLDB, vol.3, no.1, pp.1057–1067, 2010.
[21] D.Z. Wang, M.J. Franklin, M. Garofalakis, J.M. Hellerstein, and M.L. Wick, "Hybrid in-database inference for declarative information extraction," SIGMOD, pp.517–528, 2011.
[22] M. Wick, A. McCallum, and G. Miklau, "Scalable probabilistic databases with factor graphs and MCMC," PVLDB, vol.3, no.1, pp.794–804, 2010.
[23] R. Gupta and S. Sarawagi, "Creating probabilistic databases from information extraction models," VLDB, pp.965–976, 2006.
[24] R. Jampani, L.L. Perez, F. Xu, C. Jermaine, M. Wu, and P.J. Haas, "MCDB: A Monte Carlo approach to managing uncertain data," SIGMOD, pp.687–700, 2008.
[25] Z. Cai, Z. Vagena, L. Perez, S. Arumugam, P.J. Haas, and C. Jermaine, "Simulation of database-valued Markov chains using SimSQL," SIGMOD, pp.637–652, 2013.
[26] 宮崎　純，"新世代のデータベース管理システムのアーキテクチャ，" 信学誌，vol.94, no.2, pp.130–135, Feb. 2011.
[27] S. Sakr, A. Liu, and A.G. Fayoumi, "The family of MapReduce and large-scale data processing systems," ACM Comput. Surv., vol.46, no.1, 2013.
[28] C. Doulkeridis and K. Nørvåg, "A survey of large-scale analytical query processing in MapReduce," VLDB Journal. (accepted for publication).
[29] A. Pavlo, E. Paulson, A. Rasin, D.J. Abadi, D.J. DeWitt, S. Madden, and M. Stonebraker, "A comparison of approaches to large-scale data analysis," SIGMOD, pp.165–178, 2009.
[30] M. Stonebraker, D. Abadi, D.J. DeWitt, S. Madden, E. Paulson, A. Pavlo, and A. Rasin, "MapReduce and parallel DBMSs: Friends or foes?," CACM, vol.53, no.1, pp.64–71, 2010.
[31] C. Olston, B. Reed, U. Srivastava, R. Kumar, and A. Tomkins, "Pig Latin: A not-so-foreign language for data processing," SIGMOD, pp.1099–1110, 2008.
[32] A. Thusoo, J.S. Sarma, N. Jain, Z. Shao, P. Chakka, N. Zhang, S. Anthony, H. Liu, and R. Murthy, "Hive - A petabyte scale data warehouse using Hadoop," ICDE, pp.996–1005, 2010.
[33] B. Chattopadhyay, L. Lin, W. Liu, S. Mittal, P. Aragonda, V. Lychagina, Y. Kwon, and M. Wong, "Tenzing: A SQL implementation on the MapReduce framework," PVLDB, vol.4, no.12, pp.1318–1327, 2011.

[34] J.B. Buck, N. Watkins, J. LeFevre, K. Ioannidou, C. Maltzahn, N. Polyzotis, and S.A. Brandt, "Sci-Hadoop: Array-based query processing in Hadoop," Supercomputing, Article No.66, 2011.
[35] S. Melnik, A. Gubarev, J.J. Long, G. Romer, S. Shivakumar, M. Tolton, and T. Vassilakis, "Dremel: Interactive analysis of Web-scale datasets," PVLDB, vol.3, no.1, pp.330–339, 2010.
[36] A. Hall, O. Bachmann, R. Büssow, S. Gănceanu, and M. Nunkesser, "Processing a trillion cells per mouse click," PVLDB, vol.5, no.11, pp.1436–1446, 2011.
[37] 合田和生，豊田正史，喜連川優，"アウトオブオーダ型データベースエンジン OoODE の試作とその実行挙動," データ工学と情報マネジメントに関するフォーラム（DEIM），2013.
[38] A. Abouzeid, K. Bajda-Pawlikowski, D. Abadi, A. Silberschatz, and A. Rasin, "HadoopDB: An architectural hybrid of MapReduce and DBMS technologies for analytical workloads," PVLDB, vol.2, no.1, pp.922–933, 2009.
[39] E. Friedman, P. Pawlowski, and J. Cieslewicz, "SQL/MapReduce: A practical approach to self-describing, polymorphic, and parallelizable user-defined functions," PVLDB, vol.2, no.2, pp.1402–1413, 2009.
[40] C. Yang, C. Yen, C. Tan, and S. Madden, "Osprey: Implementing MapReduce-style fault tolerance in a shared-nothing distributed database," ICDE, pp.657–668, 2010.
[41] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M.J. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," USENIX NSDI, pp.15–28, 2012.
[42] R.S. Xin, J. Rosen, M. Zaharia, M.J. Franklin, S. Shenker, and I. Stoica, "Shark: SQL and rich analytics at scale," SIGMOD, pp.13–24, 2013.
[43] B. Huang, S. Babu, and J. Yang, "Cumulon: Optimizing statistical data analysis in the cloud," SIGMOD, pp.1–12, 2013.
[44] H.-C. Yang, A. Dasdan, R.-L. Hsiao, and D.S. Parker, "Map-Reduce-Merge: Simplified relational data processing on large clusters," SIGMOD, pp.1029–1040, 2007.
[45] D. Jiang, A.K.H. Tung, and G. Chen, "Map-Join-Reduce: Toward scalable and efficient data analysis on large clusters," TKDE, vol.23, no.9, pp.1299–1311, 2011.
[46] F.N. Afrati and J.D. Ullman, "Optimizing joins in a Map-Reduce environment," EDBT, pp.99–110, 2010.
[47] S. Blanas, J.M. Patel, V. Ercegovac, J. Rao, E.J. Shekita, and Y. Tian, "A comparison of join algorithms for log processing in MapReduce," SIGMOD, pp.975–986, 2010.
[48] Y. Bu, B. Howe, M. Balazinska, and M.D. Ernst, "HaLoop: Efficient iterative data processing on large clusters," PVLDB, vol.3, no.1, pp.285–296, 2010.
[49] D. Logothetis, C. Olston, B. Reed, K.C. Webb, and K. Yocum, "Stateful bulk processing for incremental analytics," ACM Symp. on Cloud Computing (SoCC), pp.51–62, 2010.
[50] M. Onizuka, H. Kato, S. Hidaka, K. Nakano, and Z. Hu, "Optimization for iterative queries on MapReduce," PVLDB, vol.7, no.4, pp.241–252, 2013.
[51] F.N. Afrati, V. Borkar, M. Carey, N. Polyzotis, and J.D. Ullman, "Map-Reduce extensions and recursive queries," EDBT, pp.1–8, 2011.
[52] J. Dittrich, J.-A. Quiané-Ruiz, A. Jindal, Y. Kargin, V. Setty, and J. Schad, "Hadoop++: Making a yellow elephant run like a cheetah (without it even noticing)," PVLDB, vol.3, no.1, pp.518–529, 2010.
[53] J. Dittrich, J.-A. Quiané-Ruiz, S. Richter, S. Schuh, A. Jindal, and J. Schad, "Only aggressive elephants are fast elephants," PVLDB, vol.5, no.11, pp.1591–1602, 2011.

（平成 25 年 11 月 22 日受付）

石川　佳治　（正員：シニア会員）

1989 筑波大学第三学群情報学類卒．1994 同大学大学院博士課程工学研究科単位取得退学．同年奈良先端科学技術大学院大学助手．1999 筑波大学電子・情報工学系講師．2004 同助教授．2006 名古屋大学情報連携基盤センター教授．2013 同大学大学院情報科学研究科教授．博士（工学）（筑波大学）．データベース，データ工学に興味をもつ．日本データベース学会，情報処理学会，人工知能学会，ACM，IEEE CS 各会員．