Googleを追え！
オープンソースのクラウド基盤

2009-12-19

増田和弘

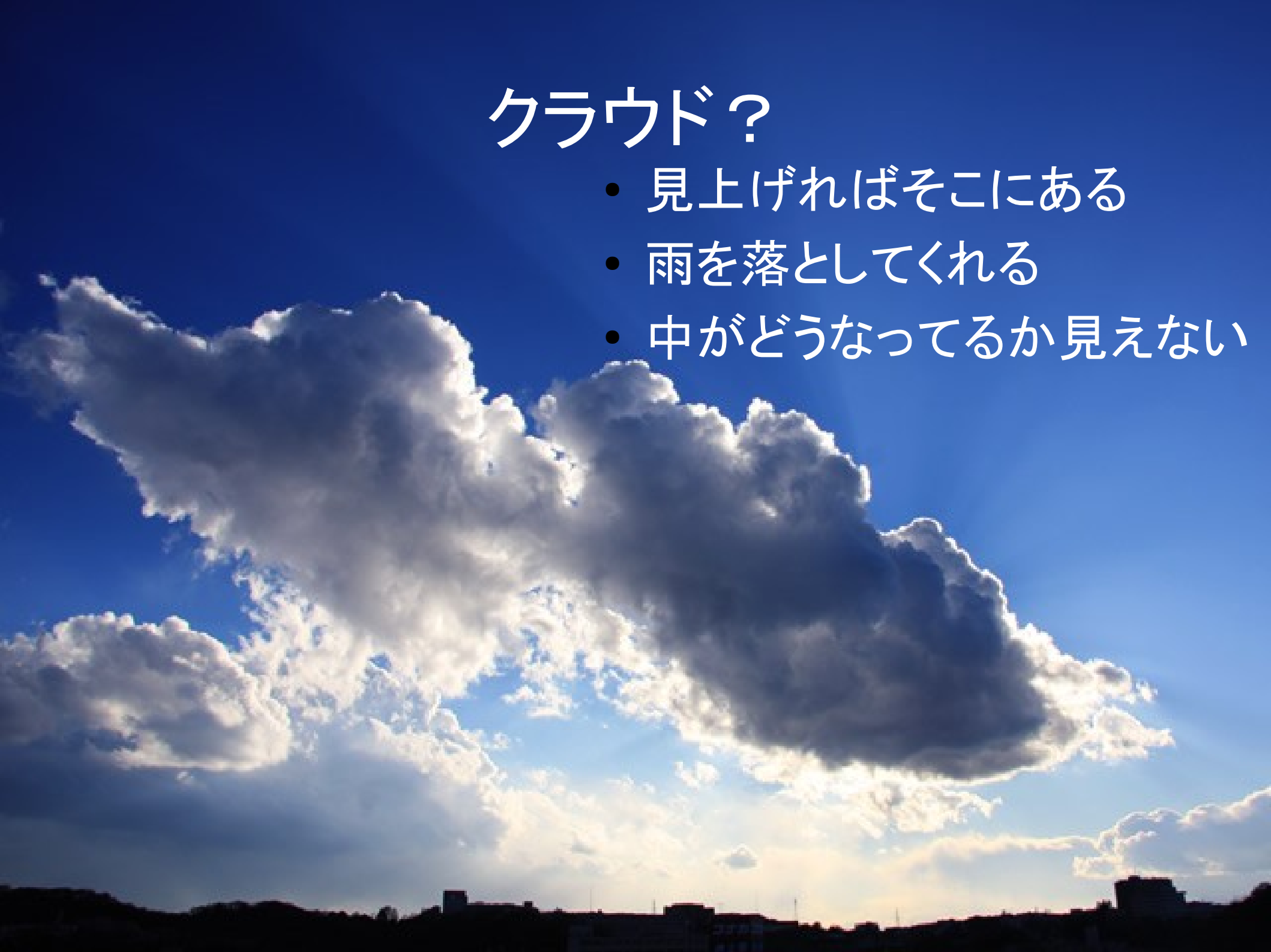
kazuhiro.masuda at justsystems.com

自己紹介 & バックグラウンド

- 1960年生まれ
- 組込系、PC用DBMS、PC通信、全文検索など
- 業務で最初に使ったコンピュータ
  - SEIKO 9500
  - マルチCPU(8086+8087+8088+8088)
- オープンソースへの貢献はとくになし
  - ユーザ or ウォッチャー
    - ハックしていません
  - Linux, PostgreSQL, Firebird
    - 社内利用・布教

内容概要

- MapReduce (Hadoop Core)
- Bigtable(Hadoop Hbase)
- Amazon AWS(Eucalyptus)


クラウド？
• 見上げればそこにある
• 雨を落としてくれる
• 中がどうなってるか見えない

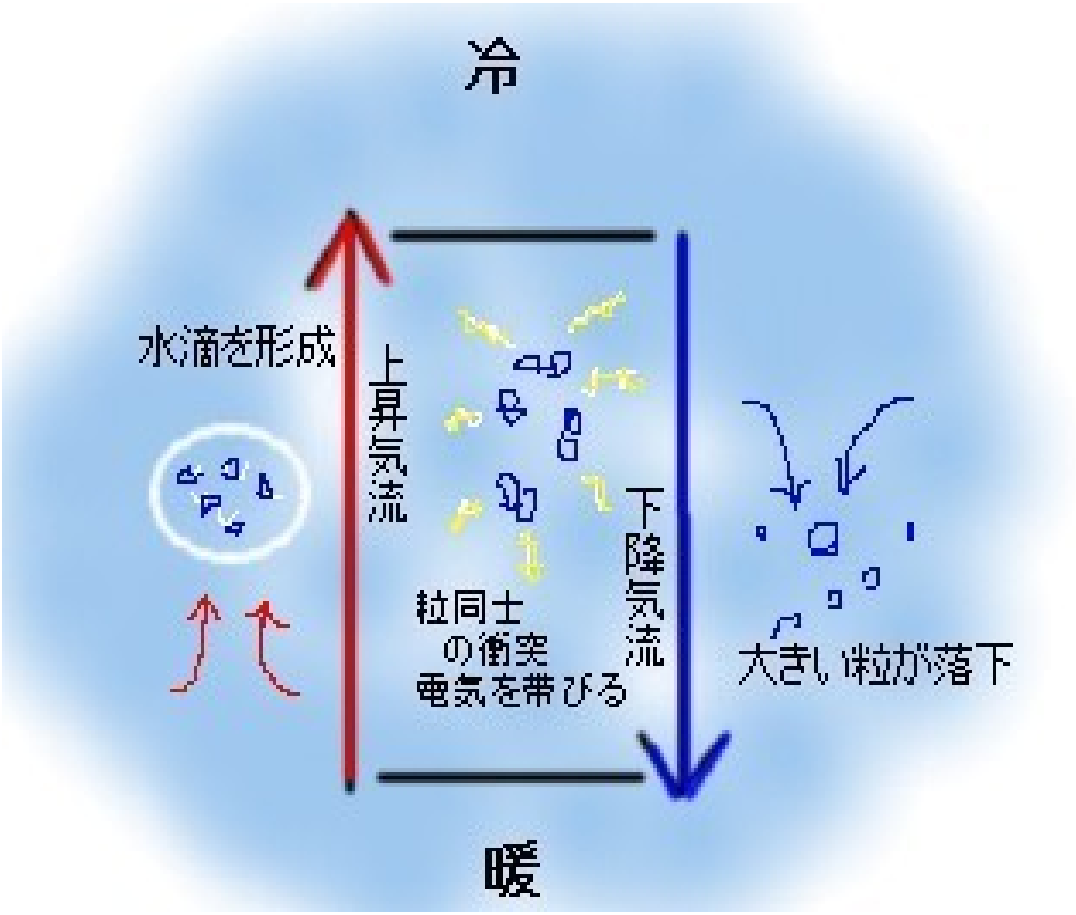
クラウドの中が見えないと…

ひどい誤解の例

「Amazon」や「Google」を「IT企業」と呼ぶのはおかしい

IntelやCiscoの技術・製品を利用して、サービスを提供しているだけじゃないか

今からクラウドの中の話



これは本当の雲

クラウドの実体
野ざらしコンテナ型データセンター

コンテナあたり
- 1000-2000サーバ
- PB級のストレージ

Googleの独自性

- サービスを提供するために、数十万台単位のサーバが稼働するデータセンターを自ら構築
- 「Webスケール」のデータを扱うためのミドルウェアも
  - 商用RDBMSでは手に負えない
  - あるいはライセンス料が非現実的

Webスケール

- 「規模」についての最近の表現
  - ユーザ数にも使うが、
  - ここでは総データ量
- GoogleEarth(2006) で 70TB
- GoogleCache(2006)で800TB

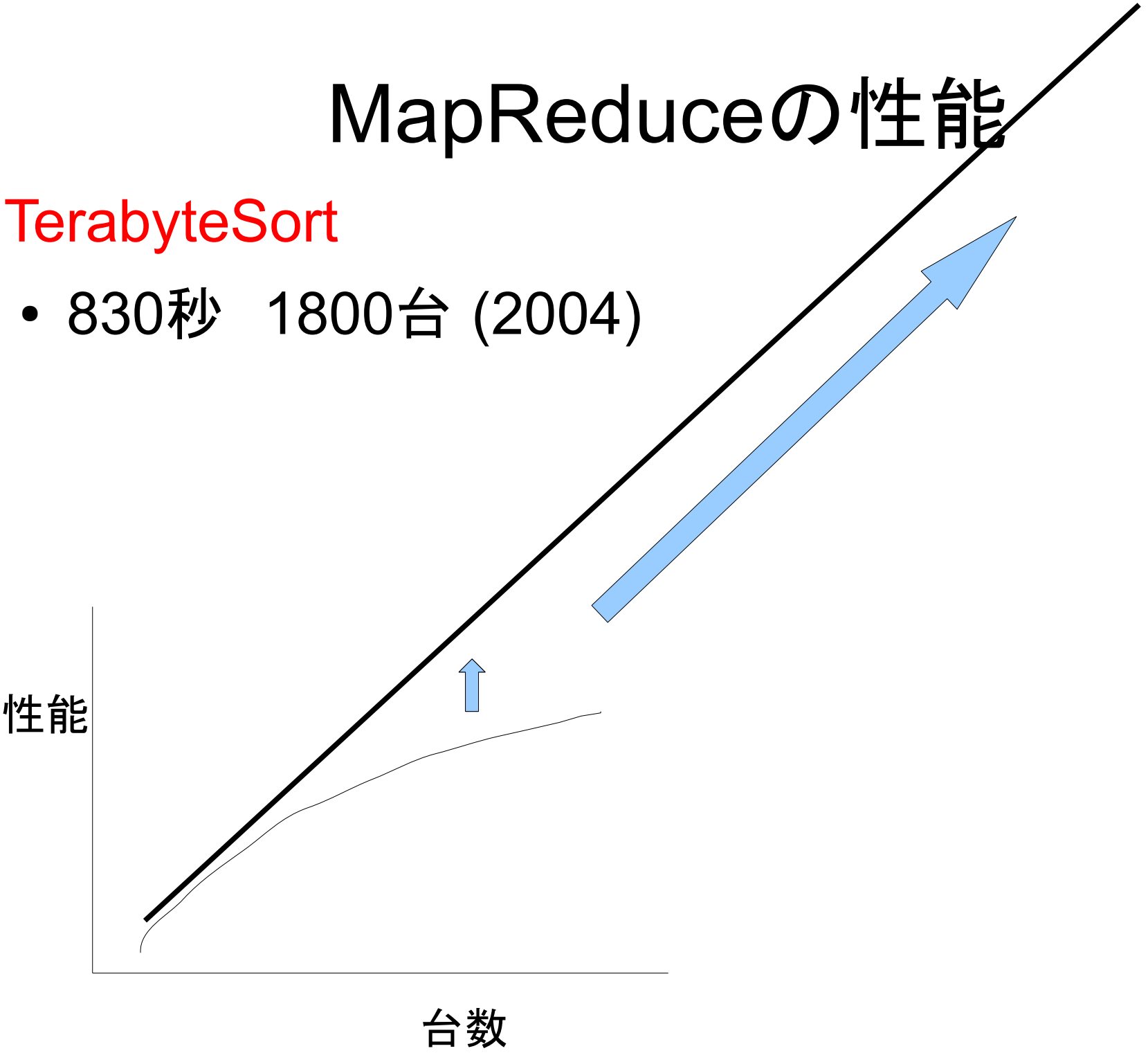
MapReduce

- J.Dean@GoogleのMapReduce論文
  - MapReduce：Simplified Data Processing on Large Clusters(2004)
- 真のスケーラビリティ
  - 100台、1000台、4000台
- コンピュータの仕事の地平拡大
  - コンピュータ技術者の仕事も拡大

MapReduceの性能

TerabyteSort

- 830秒　1800台 (2004)



MapReduceの性能(2008)

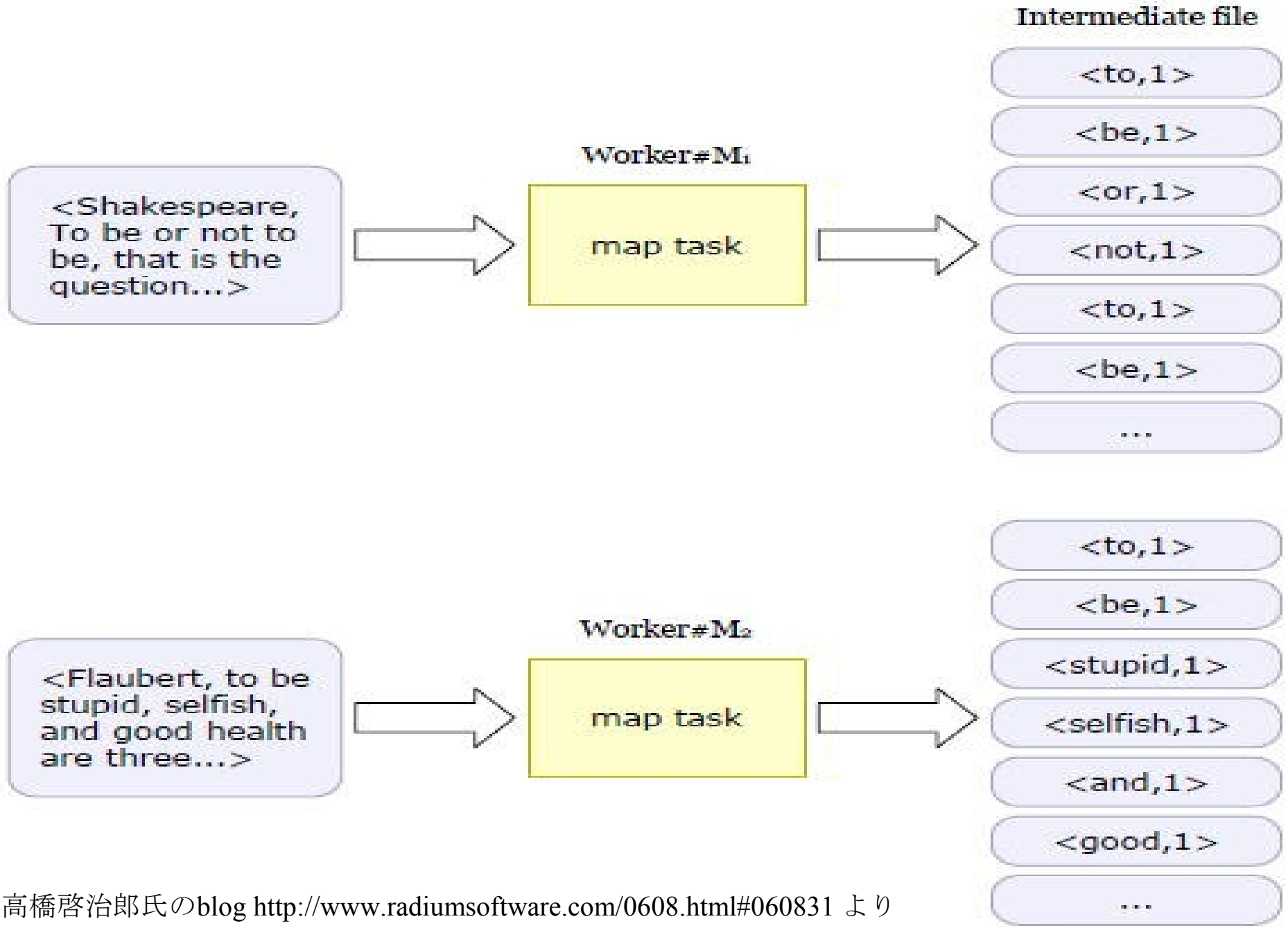
TerabyteSort

- 830秒　1800台 (2004)
- 68秒　　1000台 (2008)

PetabyteSort

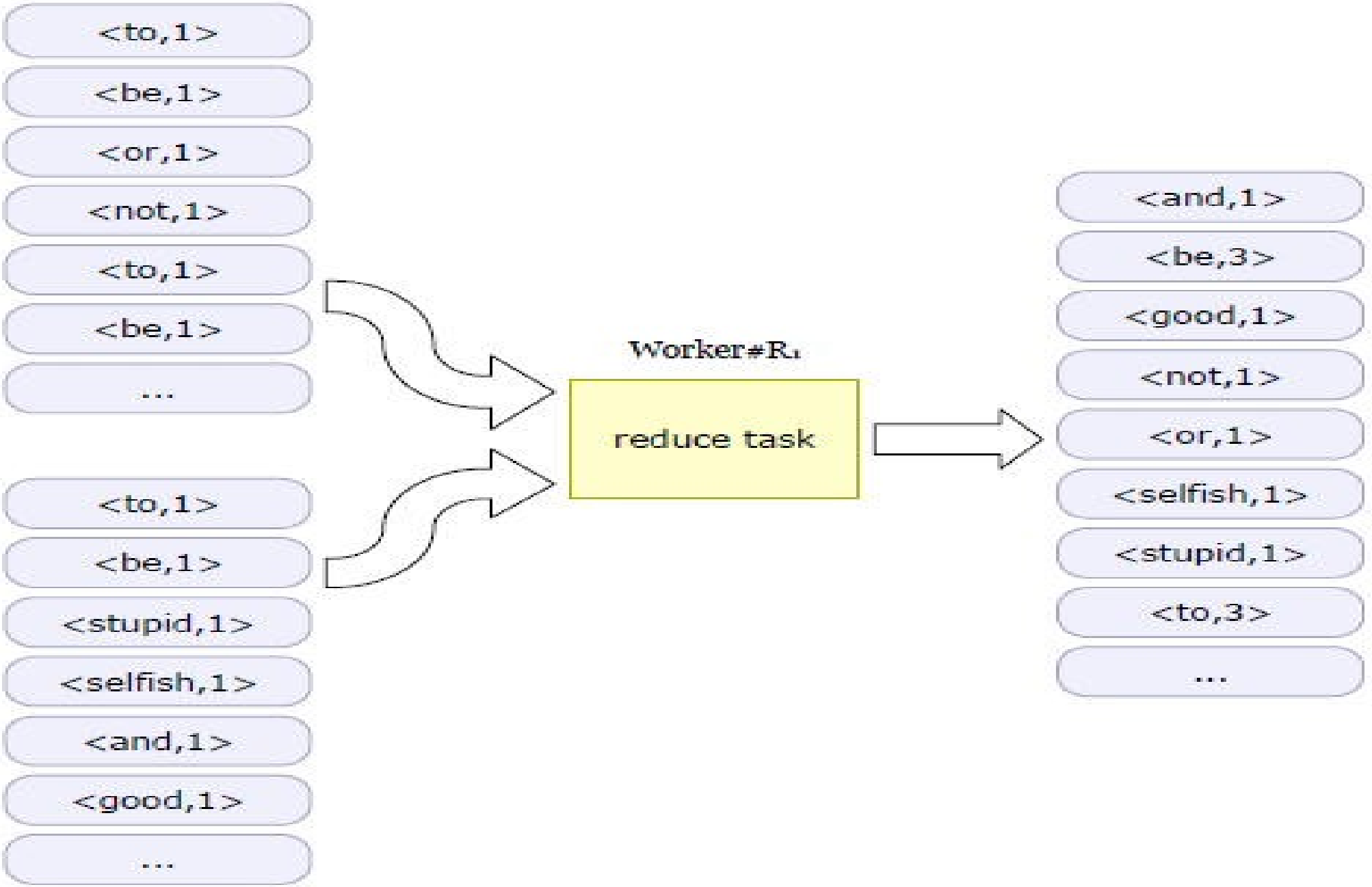
- 6時間　4000台
  - TeraByteの1000倍のデータ
  - 処理コスト（時間×CPU)は1400倍
  - O(NlogN)の世界のO(log1000)がわずか1.4

WordCountの”Map”

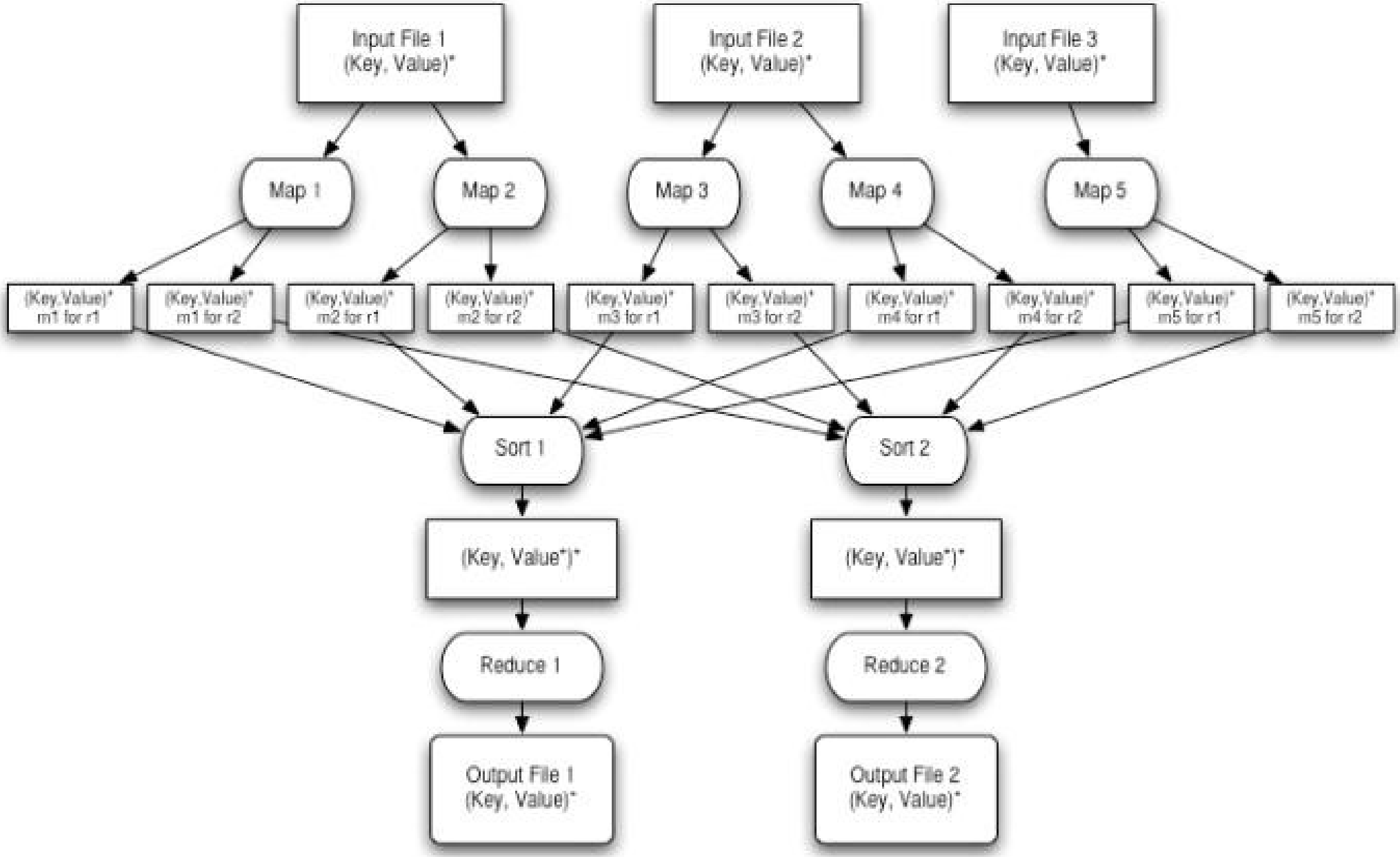


高橋啓治郎氏のblog http://www.radiumsoftware.com/0608.html#060831 より

WordCountの”Reduce”



複数のReduce処理



プログラムモデル

Map: <key1, value1> → <key2, value2>

Reduce: <key2, value2> → <key3, value3>

順序比較可能なkeyと何かの値valueの組<key,value>を、変換・集約して、有用な出力を得ようとするもの。

WordCountの場合なら、

Map: <文番号, 文内容>

→ [<単語, "1">, <単語, "1">, ...]

Reduce: <単語, ["1", "1", ...]>

→ <単語, カウント値>

Hadoop

Hadoop

- http://hadoop.apache.org/
- 並列分散処理のフレームワーク
  - 手本：MapReduce
- 分散ファイルシステム
  - 手本：GoogleFileSystem
  - DFS(DistributedFileSystem)
- Java実装のオープンソースプロダクト
- 今ではSub project多数

Hadoop Cluster @ Yahoo!(2007)

Hadoop Subprojects

- Hadoop Common: The common utilities that support the other Hadoop subprojects.
- Avro: A data serialization system that provides dynamic integration with scripting languages.
- Chukwa: A data collection system for managing large distributed systems.
- HBase: A scalable, distributed database that supports structured data storage for large tables.
- HDFS: A distributed file system that provides high throughput access to application data.
- Hive: A data warehouse infrastructure that provides data summarization and ad hoc querying.
- MapReduce: A software framework for distributed processing of large data sets on compute clusters.
- Pig: A high-level data-flow language and execution framework for parallel computation.
- ZooKeeper: A high-performance coordination service for distributed applications.

サンプルコード

- WordCount
  - Hadoop同梱のexamplesより

MapReduceの生産性

- 冗長性確保・タスク分割・タスク管理など分散処理特有の雑事からプログラマを解放
  - 1-4000倍にScale-outするWordCountが50行
- Googleで、最初のMapReduceライブラリがリリースされて1年半で900本のアプリケーションが稼働
- プログラミングモデルを1本化したことで、マージソートやノード内ソートなど共通処理となる部分を、ミドルウェア側に取り込み、その守備範囲が広がった

Hadoop DFS(GFS)

- 分散ファイルシステム
  - HDD1台にもPC1台にも入りきらない巨大なデータを多数のPCで分割して持ち合い
- 冗長性・信頼性確保
  - 通常3箇所に同じデータを保存
  - 1つを故障で失えば、他のPCにコピーを作って冗長度3を保持する
- 順アクセスに性能焦点
  - 最初から最後までRead、最後に追加するWrite
  - ブロックサイズは64/128MBと大きい
  - 非POSIX

MapReduce on DFSの性能(1)

なぜSequenceFileだけなのか？

性能要因は　シーク回数 > 転送量

	各データサイズ	データ数	合計
page	10 KB/page	100 Mpage	1.0 TB
record	100 B/record	1 Grecord	0.1 TB

メディア転送測度 100MB/sec
平均シーク時間　10msec

データ全体の複製	1.1 TB /（100 MB/sec）	2 h
管理レコードをランダムに1%更新	10 msec * 1 GB * 0.01 = 100Ksec	1 day
ページをランダムに1%更新	10 msec * 100 MB * 0.01	2 h

MapReduce on DFSの性能(2)

小さすぎる処理粒度のデメリット

- オーバーヘッド増大
- データの転送とそれに伴う制御用通信

大きすぎる処理粒度のデメリット

- 終盤で稼働ノードの一部が遊んでしまう
- トラブルでTask未了時、ロストする計算量が大きい

Jobの内容と環境（稼働ノード数とその能力）に応じた最適値

GFS/HDFSの1ブロック分と一致させれば効率がいい

小休止

Bigtable

- GFS/MapReduceはバックエンド用
- フロント用には別の要件
  - 粒度の小さいデータを
  - オンラインでランダムアクセス
  - 全体のデータ量は膨大
- 例えばGoogleEarth

可能にしたのがBigtable
http://research.google.com/archive/bigtable.html

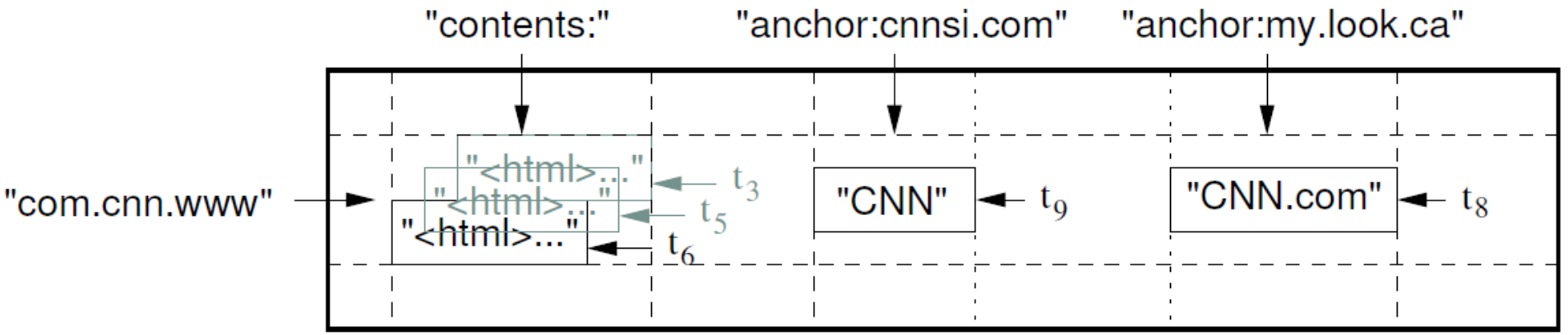
GoogleのBigtableは化物か？

- 「Webスケール」のデータ
  - GoogleEarthの画像データ 70TB(2006)
  - Google検索のWebページのキャッシュ 800TB
- 論文時点(2006)でのスループット
  - 500 client – 1800 GFS(500Bigtable node)
  - Clientは、1KB単位でread/write
  - 順読込スループット 4GB/秒　　(8K回/秒・client)
  - 順書込スループット 1GB/秒
  - ランダム書込スループット 1GB/秒

Bigtableとは(1)

- 行×列の巨大テーブルで、行に一意のキーがあり、列数は行毎に可変
- joinなし、SQLなし、複数行トランザクションなしの非RDB (Key Value Store)
- 列の値は全て可変長(LOB/BLOB)で時系列のバージョンつき

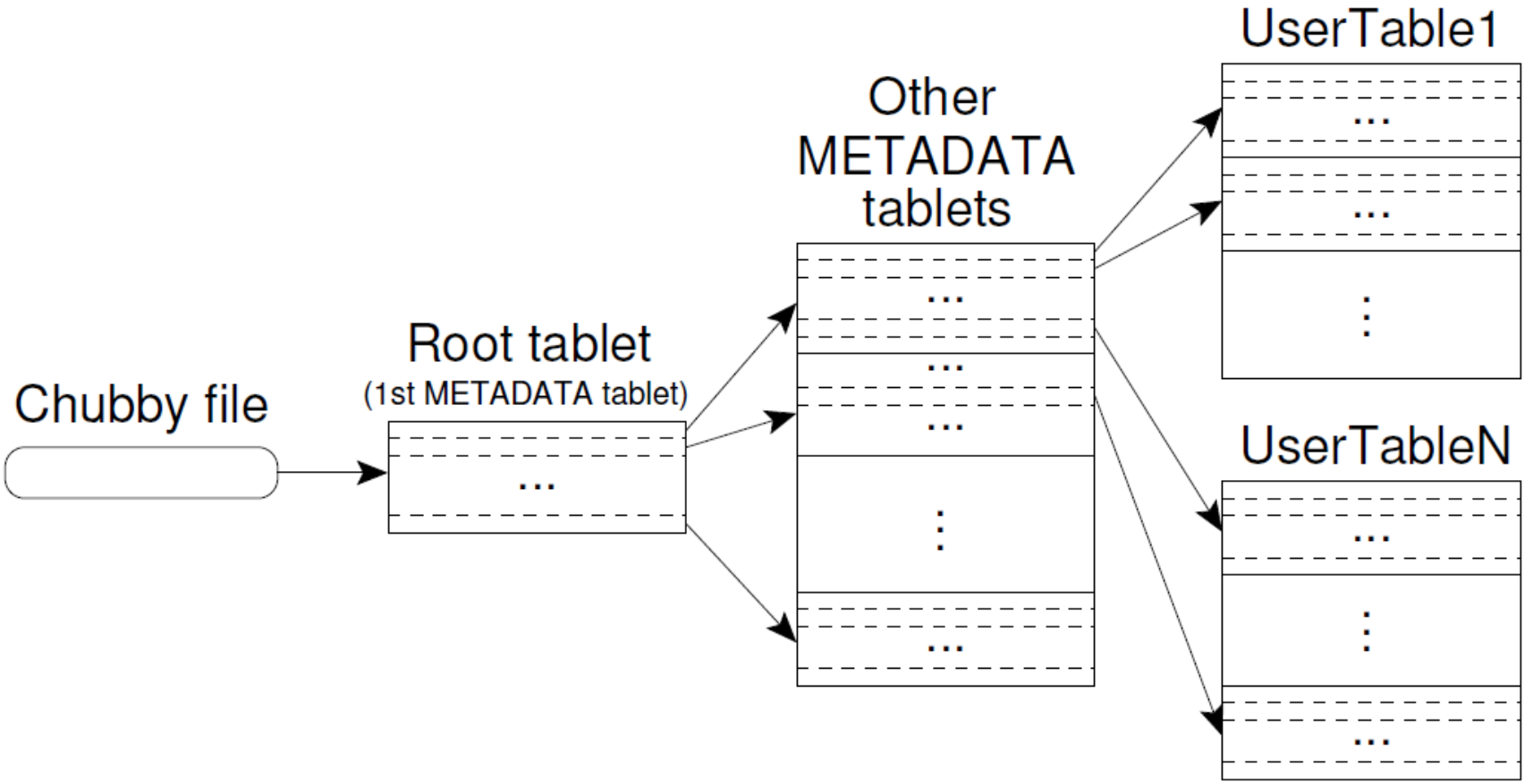
BigtableのData Model



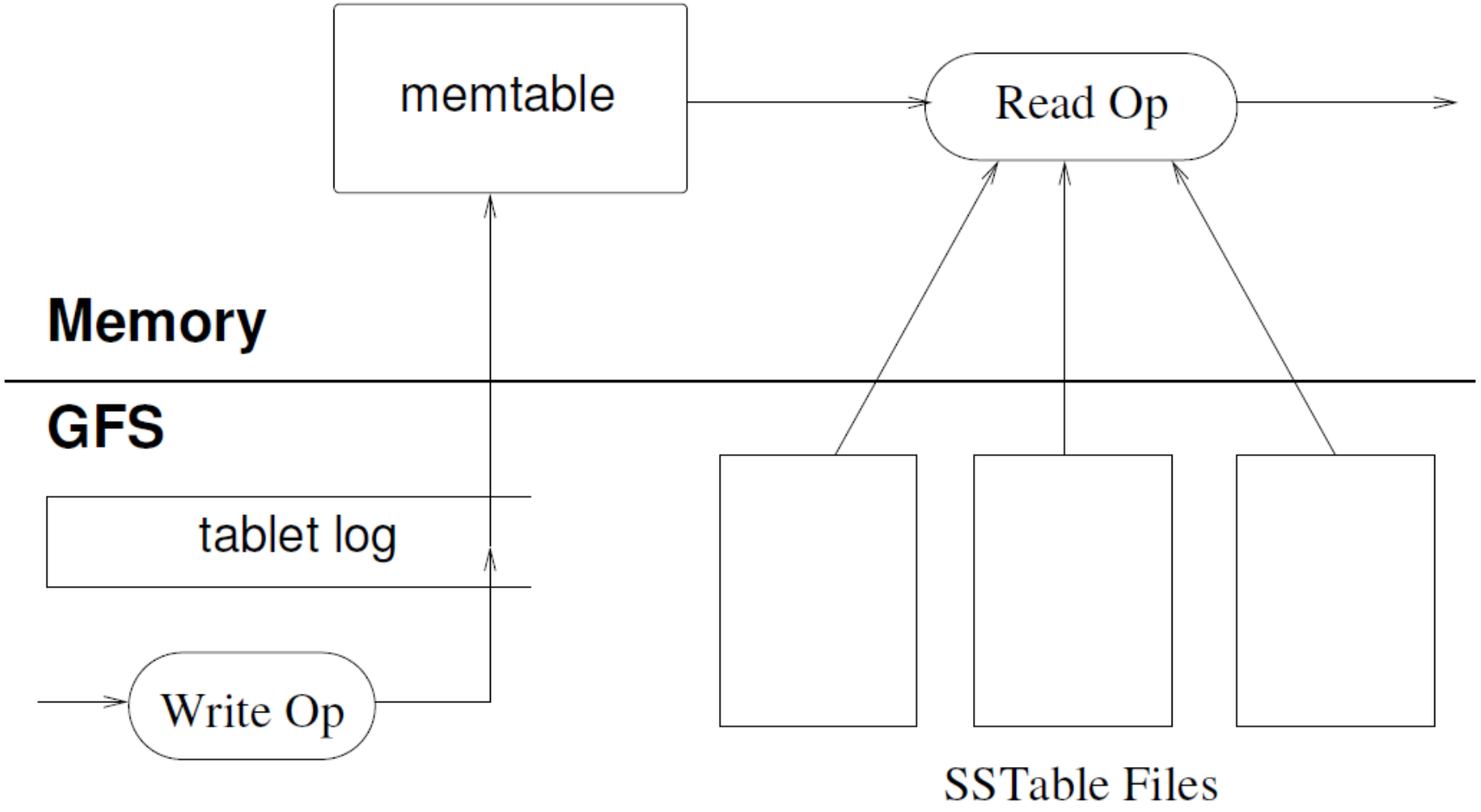
Bigtableとは(2)

- 分散処理の単位は、100MB前後でテーブルを自動水平分割したtablet
- 各serverが数十個のtabletを担当し、master-serverがtabletの分割併合割当を指示
- データ圧縮、列に関する垂直分割など、利用者側が性能に直結する部分を制御

Tabletの階層構造



File:一括作成-ReadOnly



Minor Compaction

- Tablet上のcontents更新がたまる
- メモリ上のmemtableが大きくなる
- memtableから新規のSSTableを作成
- SSTableを古い順に読んでmemtableを作れば最新イメージになる

Major Compaction

- 何度もMinor Compactionすると
  - SSTableの数が多くなる
  - 最初のmemtable作るのに手間がかかる
- 古いcontents、削除済みのcontentsを含む既存SSTable群を全部捨てて、新SSTable1つにまとめる
- Tablet全体scanに相当するので、処理量大きい

Hadoop Hbase

- Google Bigtableが手本
- http://hadoop.apache.org/hbase/
- Hadoop　DFS上に構成する構造化データストレージ
- 開発履歴
  - 27 March, 2008: release 0.1.0
  - 8 August, 2008: release 0.2.0
  - 21 September, 2008: release 0.18.0
    - Hadoop Coreと番号合わせ
  - 19 November, 2009: release 0.20.2

Hadoop Hbase

- HDFSの手前にオンメモリデータベースを配置
- 巨大なTableをスライスしたTabletを各ノードが担当
- メモリでランダムアクセス性能をカバー
- MapReduceの順アクセスの入出力にも使える

サンプルコード

- SampleUploader
  - HBase同梱のexamplesより

GFS/Bigtableの速さ(1)

- 必要な機能だけ実装して、資源を集中
  - POSIX互換なし、 SQL互換なし
  - BigtableはRDBのACID保証せず。
    - 可用性とスループット重視。
- HDD シークの削減
  - 追記のみ　一気読み、一気書き中心
- 通信量・通信回数の削減
  - Shared Nothing の並列処理
  - ファイル排他制御の削減
    - ファイル書きこみがないと排他制御も減る

GFS/Bigtableの速さ(2)

- メモリ効率の追求で、利用頻度の高いものを常駐
- バランス
  - METADATAが3階層までと実質固定
  - Tabletサイズは、B木のように一定範囲(100-200MB)を保つ
  - tablet serverの負荷は、tablet数だけでなく、実際の処理負荷もmasterが見て判断

Co-Designing

アプリケーション側も性能責任を分担

- 1行単位のトランザクションのみ
- 列方向の垂直分割(Locality group)の設定
- データ特性に合わせた圧縮ルーチンの選択

Amazon AWS

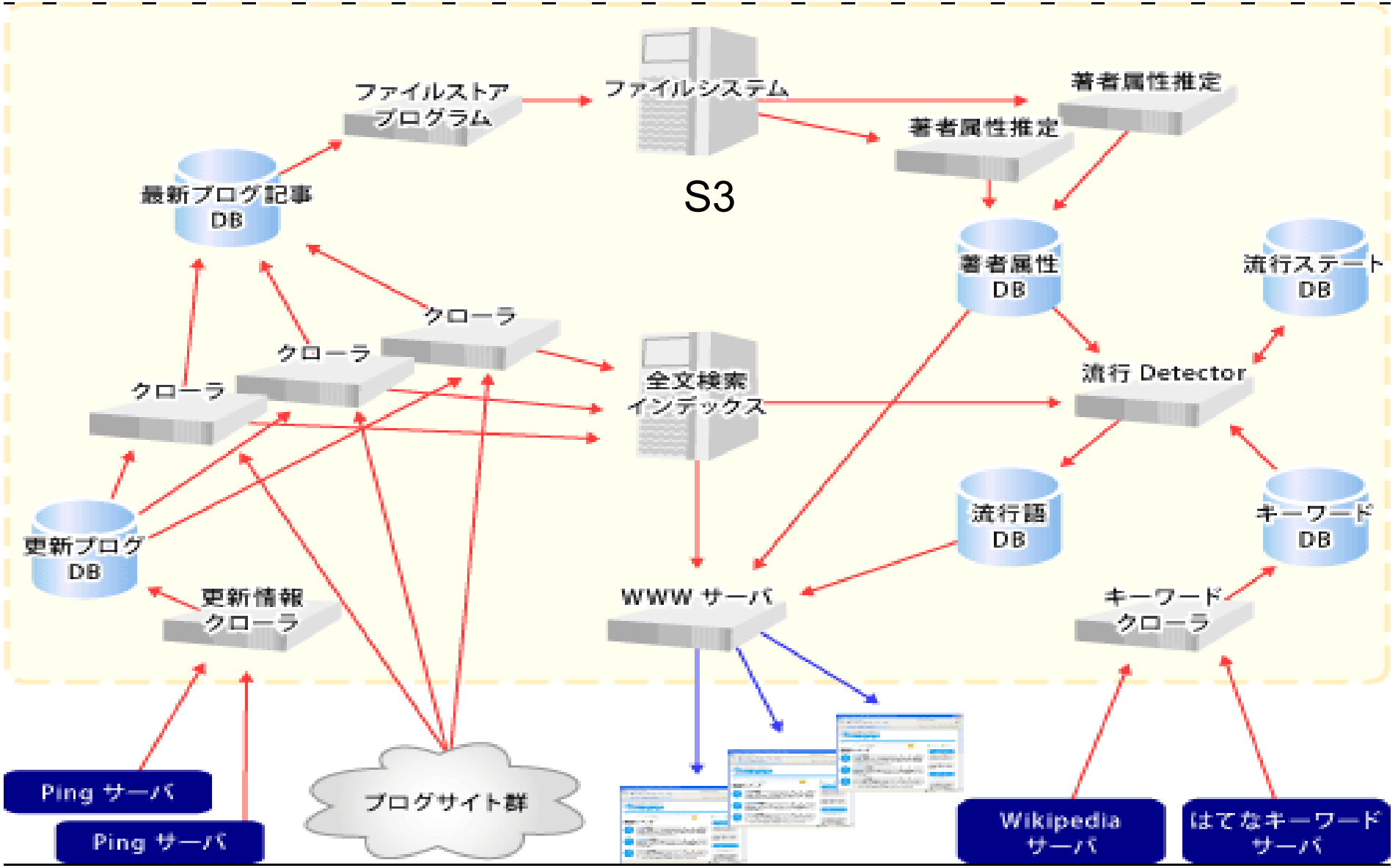
- Amazon Elastic Compute Cloud
- 仮想マシン技術を使った、Amazonの時間貸しClusterサービス
- Hadoopとアプリケーションを動かし、EBS(Elastic Block Store)にHDFSを構築可能
- EC2のインスタンスは、シャットダウンでディスクデータが消滅するが、EBSは消えない
- 料金例
  - 料金　0.10$/ 仮想ノード台数・時
  - データ転送料金 0.10 – 0.18$/GB

AWS+Hadoopのブレークスルー

- 膨大なCPUパワーの（ハード＋運用）コストが劇的に下がった
- 数百台規模で1週間だけ走らせるようなアプリケーションが可能
  - CPUコスト100台x24x7x0.10$ =1680$
  - データセンターを自前で持たなくていい
  - 弱小資本のベンチャーにも出番有り
- 比較的少ないデータを深く解析し、厳選された出力を残す応用が有望
  - 転送料、保存料を抑え、アルゴリズムで勝負

Hadoop+AmazonAWSの例

- 2007IPA未踏ソフト
- ブログを用いた「なんでも早期発見システム」
- by 大倉務さん（当時は院生）
- 日本語のブログをクロール、リアルタイムに分析
- ブログごとに著者の性別・年齢・居住地を推定
- http://blogeye.jp/



blogeyeの実装に学ぶ、Amazon EC2/S3でのHadoop活用術

http://codezine.jp/article/detail/2841

AWSのオープンソースクローン

- Eucalyptus
  - http://open.eucalyptus.com/
- Eucalyptus 調査報告書
  - クリエーションライン株式会社 (2009-6-24)
  - http://www.creationline.com/
- EC2/S3/EBS（AWS 2009-1-1と機能互換）
- 最新 Ver.1.6.1(2009-11-5)

AWS 最近のNews

- Hive(10/13)、Pig(8/11)
  - MapReduceに高級言語の皮
- Amazon Mobile Payments(10/13)
- EBS Shared Snapshots(9/24)
- Amazon Virtual Private Cloud(9/9)
- 年割値下げ
  - 年割 -34%から-44% 3年縛り -51%から-56%
- CloudFront

Hadoop/Hbaseの課題

- 性能でGoogleにキャッチアップ
  - TeraByteSortでも水が開いている
- 安定性
  - Hadoop Coreはともかく、Hbaseは最初のリリースから2年経っていない。
  - Hbaseは安定しているとはいえない。メモリも喰う。
  - APIなど仕様の安定も必要
- ユーザを増やす
  - ユーザが増えないとバグはとれない、性能チューニングも進まない
- 本家Google App Engine for Java + Bigtableが最大脅威

終わりです