丸山先生レクチャーシリーズ 2010 第3回
2010年 4月 22日(木)　楽天株式会社 本社

key-value ストアの基礎知識

首藤 一幸

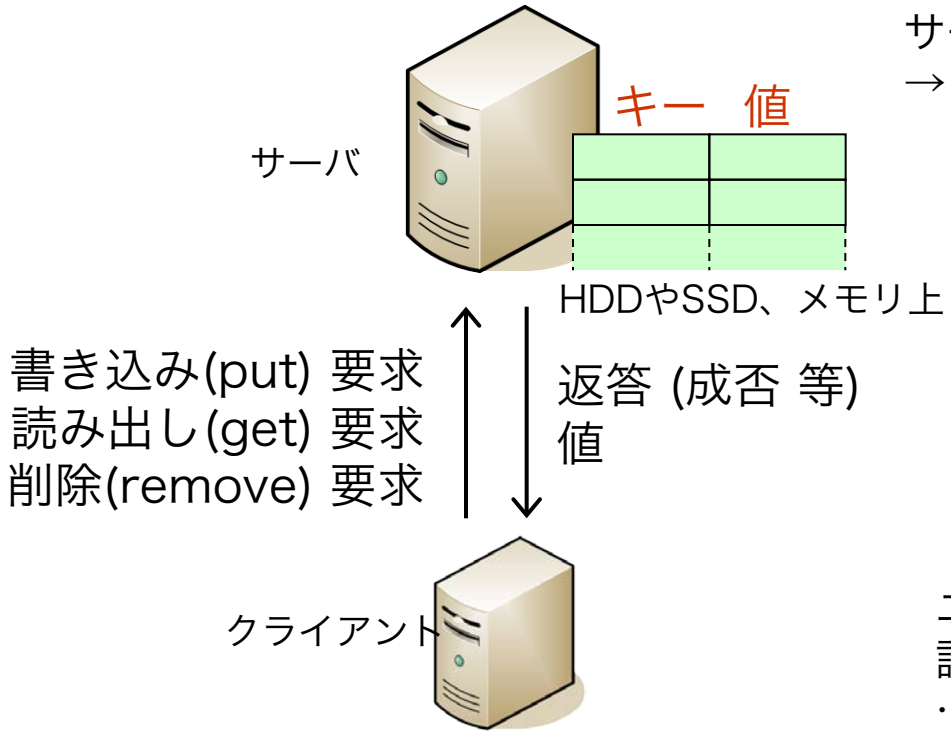
内容

- key-value ストア
- NoSQL データストア
- 特徴とその根源
- さらなるスケーラビリティを
  - デモ

- スライド 30枚

key-value ストアとは

- キーと値の組を書き込み、
  キーを指定することで値を読み出せる
  データベース管理ソフトウェア



サーバクラスタを複数台で構成
→ **分散 key-value ストア**

ユーザID：最終ログイン時刻
証券の銘柄： 株価
…

key-value ストアとは

- プログラミング言語でいうと
  連想配列, 辞書 (dictionary), map
  - 「ハッシュ表」← 実装の仕方, データ構造・アルゴリズム

Java 標準ライブラリの java.util.Map 型

```
Map aMap = new HashMap();
aMap.put(“key1”, “value1”);          // “key1” をキーとして “value1” を格納
... = aMap.get(“key1”);              // “key1” に対応する値を取得
```

Python の dictionary

```
aMap = dict()
aMap[‘key1’] = ‘value1’              # ‘key1’ をキーとして ‘value1’ を格納
... = aMap[‘key1’]                   # ‘key1’ に対応する値を取得
```

key-value ストアとは

- シンプルな問い合わせ
  - キーを１つ指定
    - ものによっては、範囲検索も。例：aで始まる語からcで始まる語まで
  - シンプル、というか貧弱
  - それを武器にして、**高い性能**と**スケーラビリティ**を達成
- ネットワーク越しにアクセス
  - クライアントライブラリを使ってアクセスする。
  - データをローカルディスクに保存する DBM は key-value ストアか？
    - 例: Berkeley DB, Tokyo Cabinet, …
  - 通常、key-value ストアとは呼ばない。

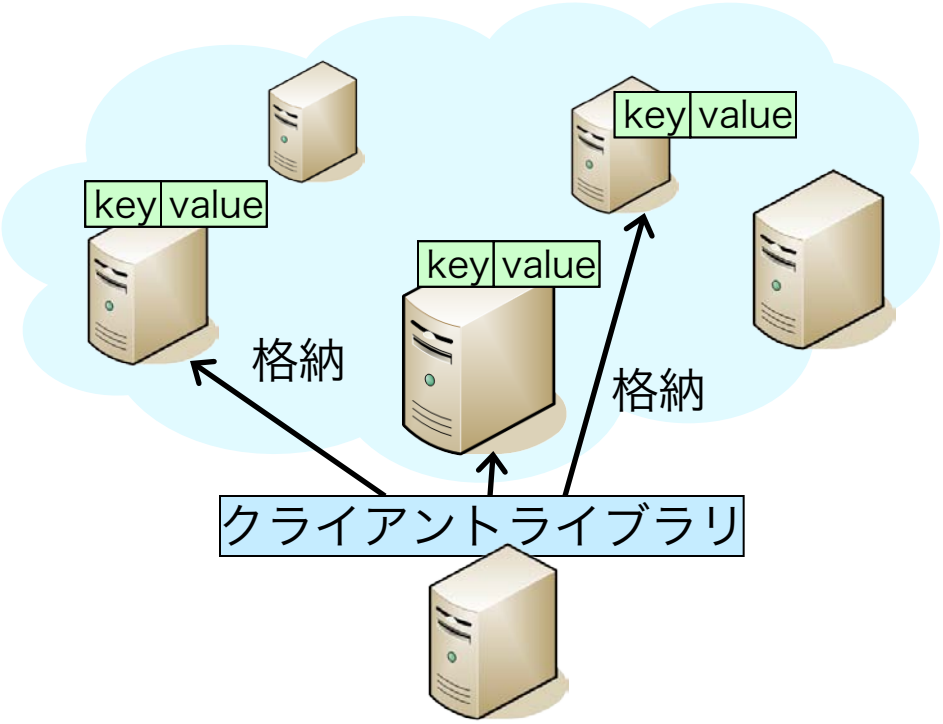
key-value ストアとは

- データの格納先
  - ソフトウェアによって、
    メモリ上だったり HDD や SSD 上だったり。
  - 格納先, ストレージエンジンを選べるものも多い。

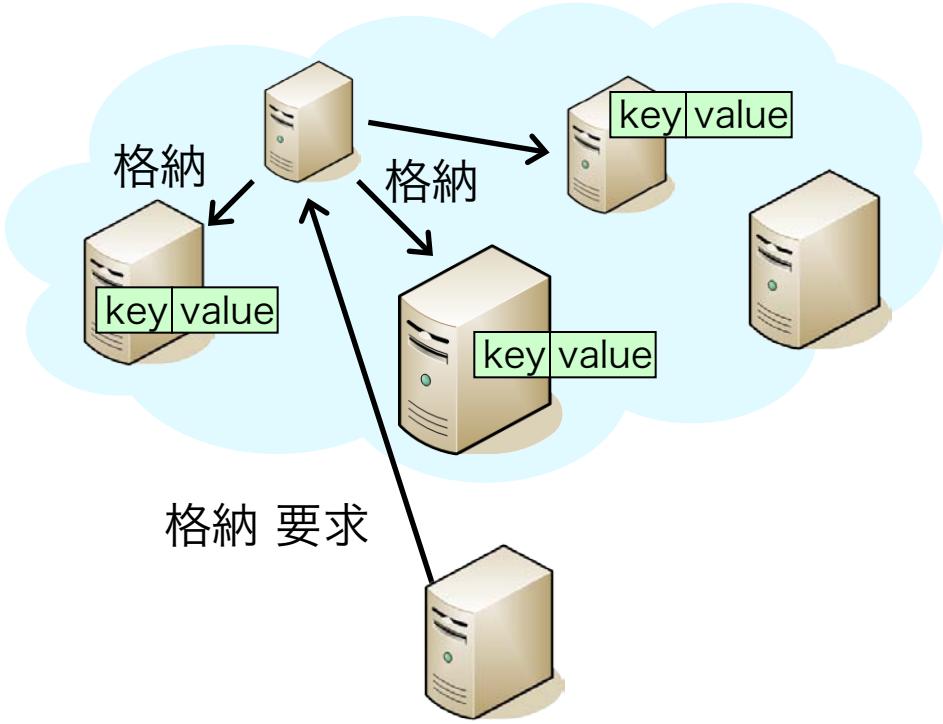
  - HDD だから遅いとも限らない。
    - UNIX 系 OS, Windows などは、空きメモリでファイルの内容をキャッシュする。
    - cf. kumofs (古橋), Tokyo Cabinet (平林)

key-value ストアとは

- 分散 key-value ストア
  - 複数台にデータを分散させられる key-value ストア。
  - 台数で性能・容量を稼ぎ、複製で耐故障性を稼ぐ。



クライアントライブラリによる
複製の作成



分散 key-value ストアでの
サーバ群の連携例

ソフトウェア

- オープンソースソフトウェア
  - Flare (藤本氏)　GREE で活用
  - Kai
  - kumofs (古橋氏)
  - memcached　Facebook 他で活用
  - ROMA　楽天で活用
  - Tokyo Cabinet / Tokyo Tyrant (平林氏)　mixi で活用
  - Voldemort (LinkedIn 社発)
- 商品
  - Coherence (Oracle)
  - Velocity (Microsoft)
  - WebSphere eXtreme Scale (IBM)

どう使われているか

- 強みが活きて、
  割り切っている点を許容できるような応用

- ○ 強み
  - RDB とは桁違いの**高性能** (qps, 低遅延)
  - 数十台以上へのスケーラビリティに支えられた**大容量**

- △ 割り切っている点
  - シンプルな問い合わせ方法
  - キー１つといった小さな単位での atomic な読み書き
  - 複製間の緩めの整合性

詳細は後ほど

高性能 狙い

- volatile なデータを対象とする
- RDB のキャッシュとして使う

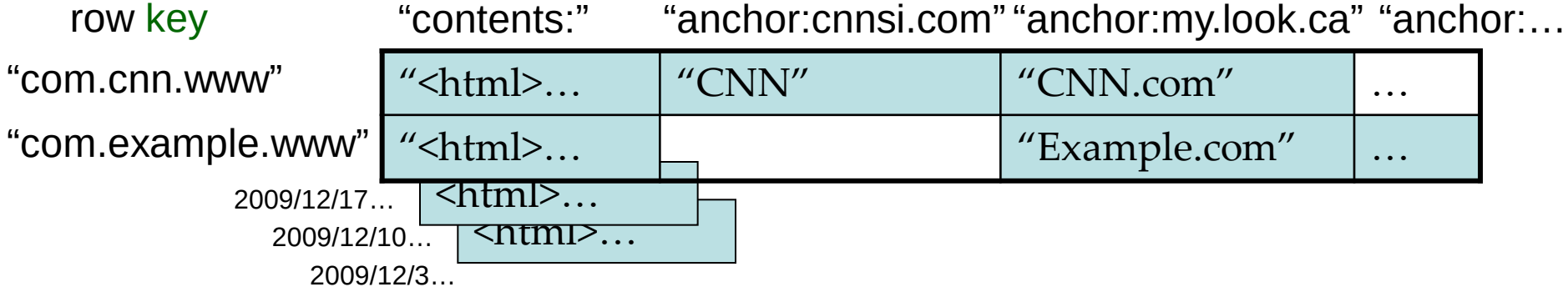
- GREE, mixi
  - アクセス履歴、というか足あと：数万 qps
- Amazon (2007年)
  - 分散 key-value ストア Dynamo を使って ショッピングカートのデータを管理
  - １日あたり、数千万の要求と300万以上のチェックアウトを処理
- Facebook (2008年12月)
  - memcached を使って RDB へのアクセスを軽減
  - 800台以上のサーバで 28 テラバイトを超えるメモリ領域

高性能 狙い

- エンタープライズでの例
  - 金融機関からの RFP「応答 10ミリ秒以内」
  - RDB だけの設計では満たせなかったところ、key-value ストアを活用することで達成
  - RDB のキャッシュ
- 実は、エンタープライズでの歴史は長い
  - Tangosol 社 Coherence → 現 Oracle社 Coherence
    - **2001年**第4四半期 リリース
  - IBM社 ObjectGrid → 現 WebSphere eXtreme Scale
    - 2005年 7月 リリース
  - memcached
    - 2003年 5月 30日 ver.1.0.0 リリース

大容量 狙い

- Google
  - ウェブから取得しまくったコンテンツを Bigtable に保存。
  - ～1千台？でクラスタを構成。



Google の基盤ソフト	機能	狙い・性質
Google File System (GFS)	ファイルを保持	大容量
Bigtable	表形式データストア	低遅延
MapReduce	データ処理	高スループット

ん？

- キーと値のペアではなくて、表だよね。
- **Bigtable は key-value ストアなのか？**

- key-value ストアだ！派の論拠
  - Bigtable 論文
    - A Bigtable is … sorted map. The map is indexed by a row key, column key, and a timestamp; each value in the map is an uninterpreted array of bytes.”
      (row:string, column:string, time:int64) → string
    - キーから値を取り出すのだから key-value ストアだろ
  - Google App Engine のウェブサイト
    - … Bigtable, a large, distributed key-vlaue store …

key-value ストアとは呼ばない派

- データモデルが key-value ペアのものに限って key-value ストアと呼ぶのがいいのでは？

- Bigtable の類 (データモデル: 表) が key-value ストアだとすると、RDB (表) も key-value ストアではないのか？
- RDB を key-valueストアに含めるとなると「key-valueストア」という語の存在意義がなくなる
  → 含めない

- Bigtable の類と RDB の間に違いはあるが、「key-value」という表現は当たらないのではないか。

みんな、非 RDB を表す言葉に飢えてて、飛びついちゃった？

Bigtable の類と RDB の違い

- 効率よい問い合わせ処理のためには、
  特定 column / 列についての条件指定が要る。
  - row を担当するサーバを特定するため

- JOIN処理はサポートしない。その代わり、
  表のスキーマが固定されていない。

- 表全体にわたる ACID 性は緩和されている。

ちょうどいい言葉が発掘されました！

NoSQL というムーブメント

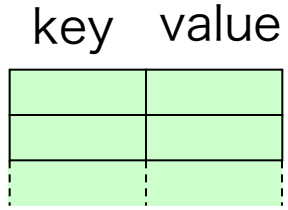
- **NoSQL** is an umbrella term for a loosely defined class of non-relational data stores that break with a long history of relational databases and ACID guarantees. Data stores that fall under this term may not require fixed table schemas, and usually avoid join operations. The term was first popularised in early 2009. (Wikipedia, 2009/12時点)

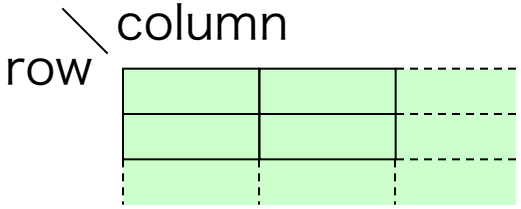
- **RDB ではないデータストア**
- 表全体に渡る ACID 性は緩和して、代わりに何かを得る。
  - ACID: データベースのトランザクションが満たすべき性質。atomicity/原子性, consistency/一貫性, isolation/独立性, durability/永続性。1970年代終わり頃に定義。
  - CAP則とか eventual consistency とか…略

NoSQL: いっぱいあります

- memcached, Bigtable, Dynamo, Amazon SimpleDB, Cassandra, Voldemort, Ringo, VPork, MongoDB, CouchDB, Tokyo Cabinet/Tokyo Tyrant, Flare, ROMA, kumofs, Kai, Redis, HadoopのHBase, Hypertable, PNUTS, Scalaris, Dynomite, ThruDB, Neo4j, IBM ObjectGrid, Oracle Coherence, Velocity, …

- データモデル
  - OSS memcached — map (key-value)
  - Amazon Dynamo — map (key-value)
  - Google Bigtable — table
  - Amazon SimpleDB — table
  - Windows Azure Table — table





NoSQL データストアと key-value ストア

- key-value ストアは、
  NoSQL データストアとして扱われたり、
  そうでなかったり。

- **性質が共通**
  - 強み：高性能, スケーラビリティ → 大容量
  - 割り切り：問い合わせ方法に制約, …

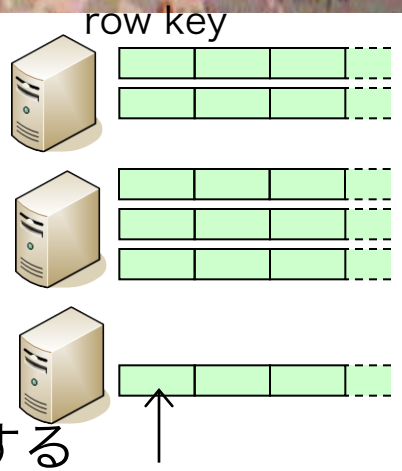
以降の話は、両者に共通

特徴の根源：
分散と atomicity との関係

atomic: 不可分な

- 1 key, 1行が基本単位
  - key-valueペア
  - 行 (row, item, entity)

の　集合



- **atomic な更新の単位 ⊆ 分散の単位**とする
  - key-value ペア, 行は、同一マシンに載る。
  - key, 行が異なると、同一マシンに載るとは限らない。

=

- **分散トランザクションは availability を損ねる**
  → 避ける

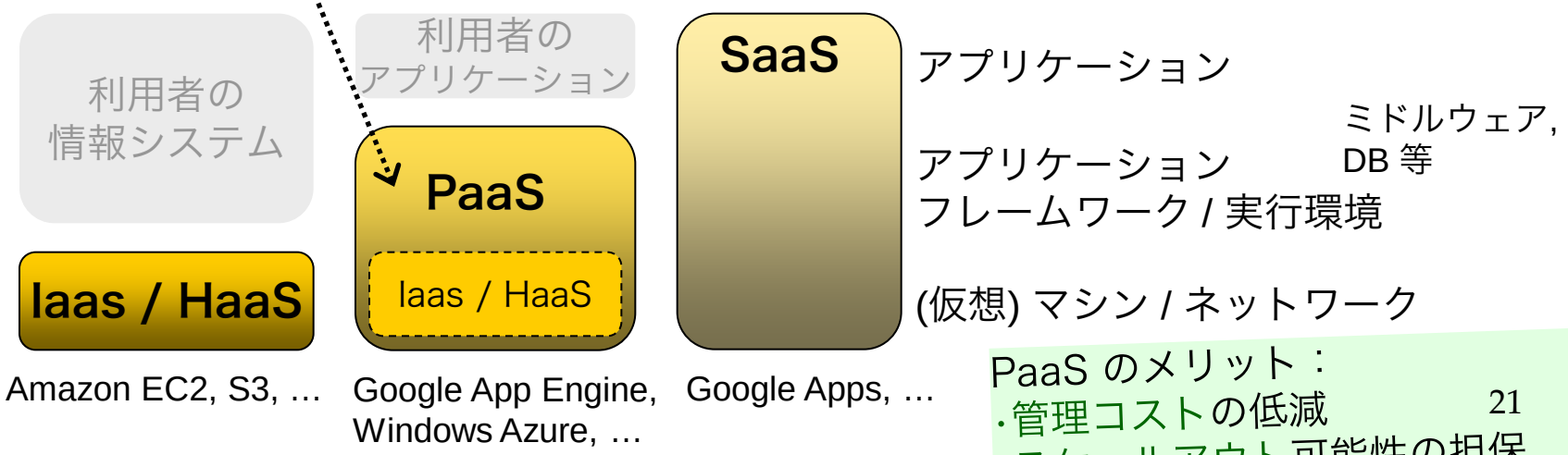
atomic な更新は
同一サーバ上の key-value ペア、行でだけ保証

特徴

- 強み
  - 高い性能
    - 水平分散 → サーバ台数を増やせる
    - atomicity 保証を同一サーバ上に限定 → スケールアウト
  - 高いスケーラビリティ → 大容量
    - 水平分散
- 割り切り
  - 効率のよい問い合わせ方法が限られる
    - 特定 column / 列についての条件指定がないと、全サーバに問い合わせる必要が生じる。
  - 細かい単位に限られた atomic な更新
  - 複製間の緩い整合性

講演概要の意味

- “クラウド、特に PaaS 向けのソフトウェア開発が現実のものとなり、そこではリレーショナルデータベースとは違ったデータベースが勢いを増しています。その代表である key-value ストアを解説します。”
- 意味
  - PaaS の一部として当たり前のように NoSQL データストアや key-value ストアが提供されている
  - Google App Engine の datastore (Bigtable) や memcache, Azure Table, Amazon SimpleDB, …



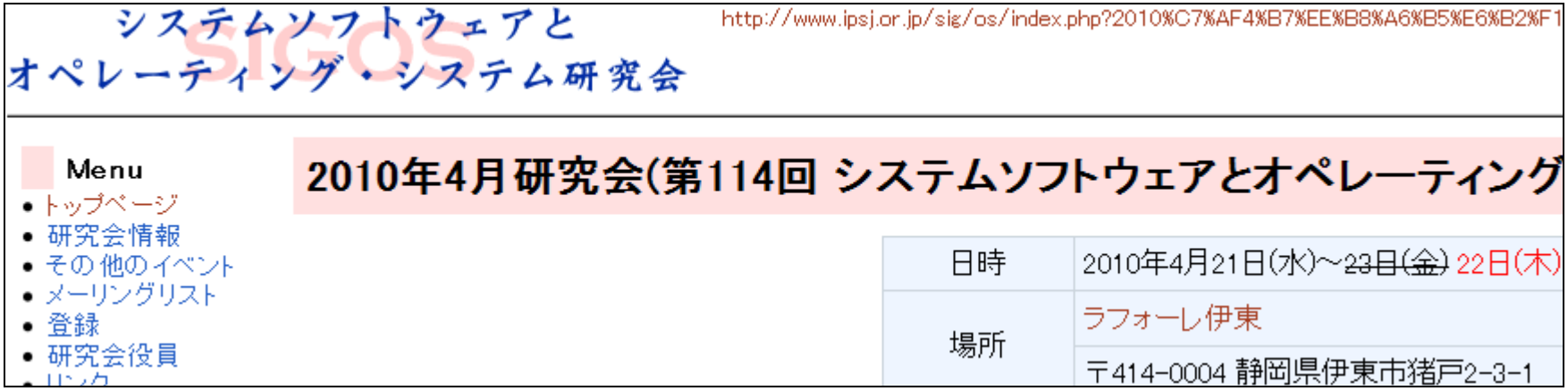
PaaS のメリット：
・管理コストの低減
・スケールアウト可能性の担保

さらなるスケーラビリティを

NoSQL / key-value ストアと
peer-to-peer のいいとこ取り

考案した新方式を本日発表

- 首藤研メンバが、今朝、発表したはず。



●4月22日（木）

■10:50-12:20 ネットワークとストレージ

(11) 柔軟な経路表によるオーバレイネットワークの設計
　　○長尾 洋也，首藤 一幸（東工大）

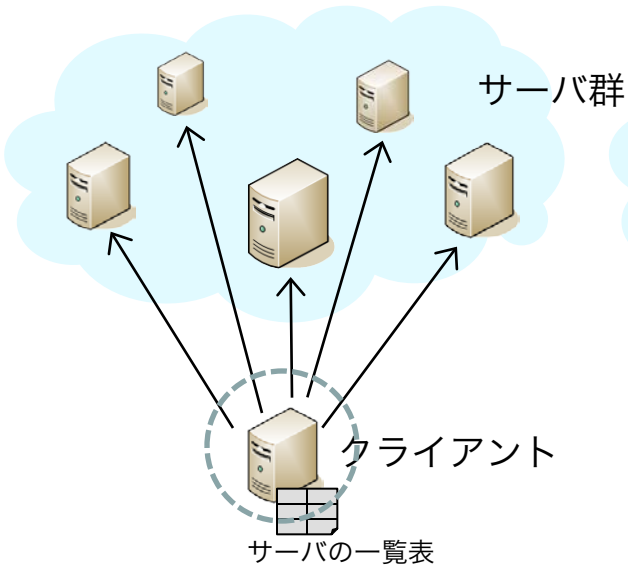
(12) 仮想マシンモニタによるきめ細かいパケットフィルタリング
　　○安積 武志（東京工業大学），光来 健一（九州工業大学），千葉 滋（東

(13) SSDをディスクキャッシュとして利用するLinuxブロックデバイスドライバ
　　○仁科 圭介，並木 美太郎（東京農工大学）

分散 key-value ストア / NoSQL データストアのアーキテクチャ

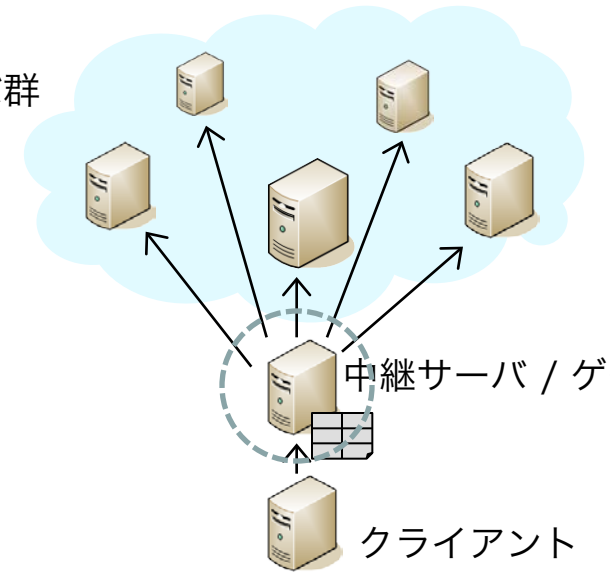
これについて考える

- 担当サーバの判断を誰が行うか？



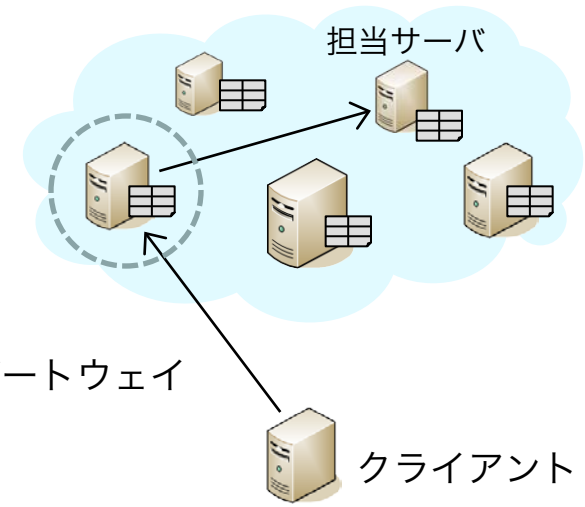
(1) クライアント側が判断する

例：memcached, ROMA, Dynamo



(2) 中継サーバが判断する

例：kumofs



(3) サーバが判断・転送する

例：Dynamo, Cassandra

それぞれの得失・向き不向きについて、かなり議論の余地がある

分散データストアが狙うスケーラビリティ

- Dynamo: Amazon の分散 **key-value ストア**
  - **数百台**まで “its initial design targets a scale of up to hundreds of storage hosts.”
    - 本当にそこまで行けるか？
      経路表 (サーバ一覧表) の一貫性維持が極めて大変。

- peer-to-peer
  …というか、非集中分散システムの技術
  DHT (分散ハッシュ表) / 構造化オーバレイネットワーク
  - **数百万**ノード以上

サーバ側

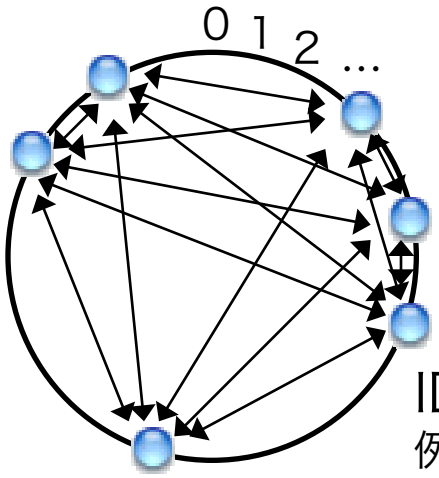
性能 重視

スケーラビリティ 重視

peer-to-peer 由来

分散データストアが狙うスケーラビリティ

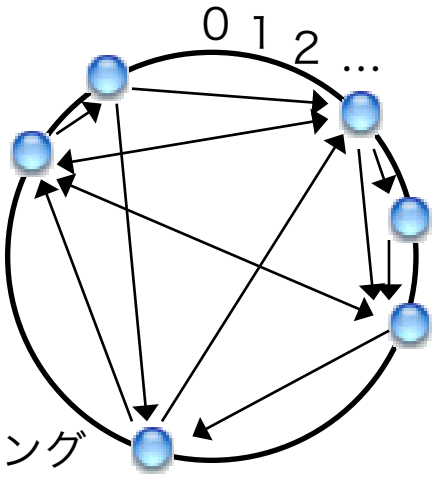
分散 key-value ストア	分散ハッシュ表 (DHT)
•サーバ側 •担当サーバに直接到達 •全サーバが全サーバを把握 　•高性能 (低遅延) •〜 数百ノード？	•peer-to-peer 由来 •中継を繰り返して到達 •一部のサーバのみを把握 　•高 スケーラビリティ •〜 数百万ノード？



全対全

ID 空間
例えば 0 〜 2^160 のリング

経路表



Overlay Weaver – デモ

http://pl.shudo.net:3998/

Get results:

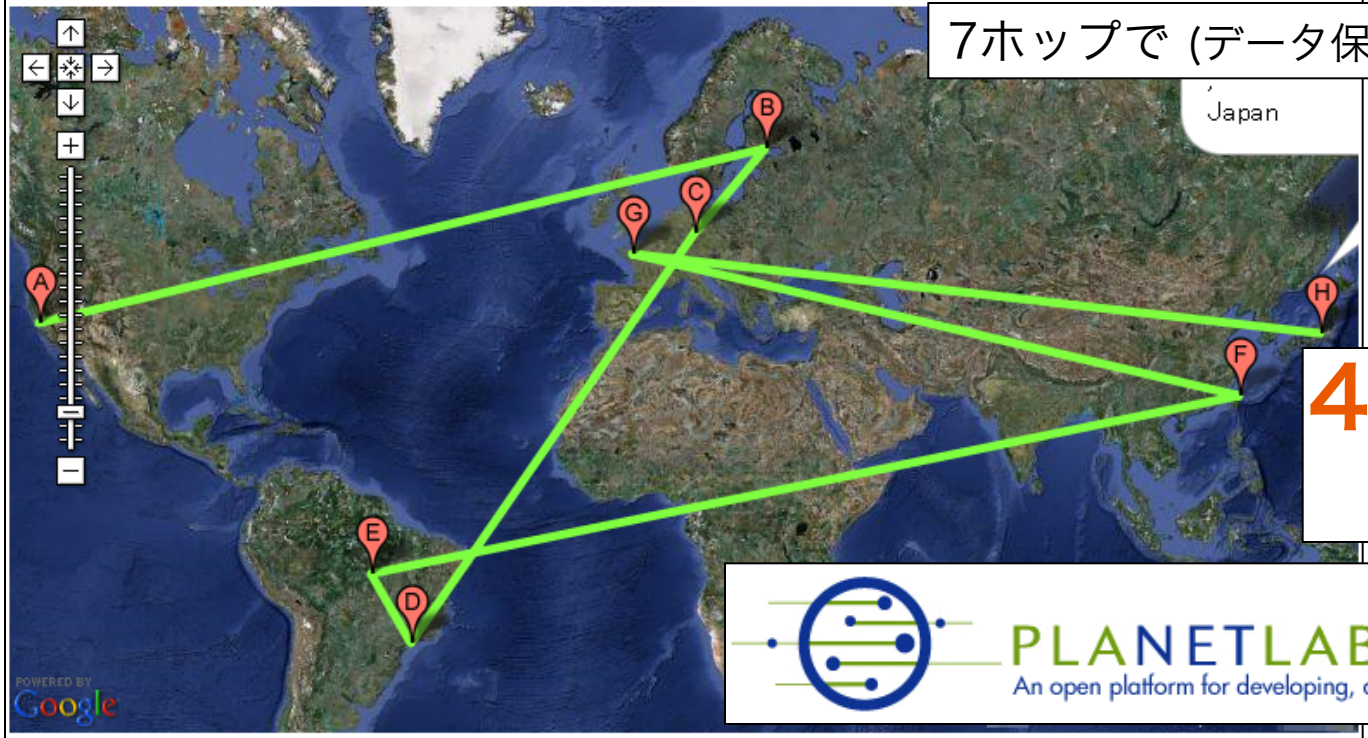
key: www.google.com
value: 66.249.89
value: 66.249.89
value: 66.249.89

Route

Hop	Node	ID	time
0	http://pl.berkeley.edu:3998/	f9aaf29cd44ef232c8fa3dfcd0b6e2da726bd59f	0
1	http://pl.vtt.fi:3998/	3b7d8a0a8d1a956a20fc8fe7da286f	
2	http://pl.tu-ilmenau.de:3998/	7ba15fb06db8821e233f94ab749765	
3	http://pl.usp.br:3998/	bc25172a17edcc0eae2ccf817120d2	
4	http://pl.pop-rs.rnp.br:3998/	ce9aaafedbed4a0d9555f5e0e8e340	
5	http://ee.ntu.edu.tw:3998/	d7d18e6ca60c618a5eb184fbdf861e	
6	http://.irisa.fr:3998/	d7ecae28f58111fd0b03eeb897dfd678504cc5a8	1345
7	http://.otemachi.wide.ad.jp:3998/	d8c7ccc7a5c6efeea96e346818918dc9ebcc4dad	1810

DHT として運用して DNS を模擬

7ホップで (データ保持) 担当ノードに到達



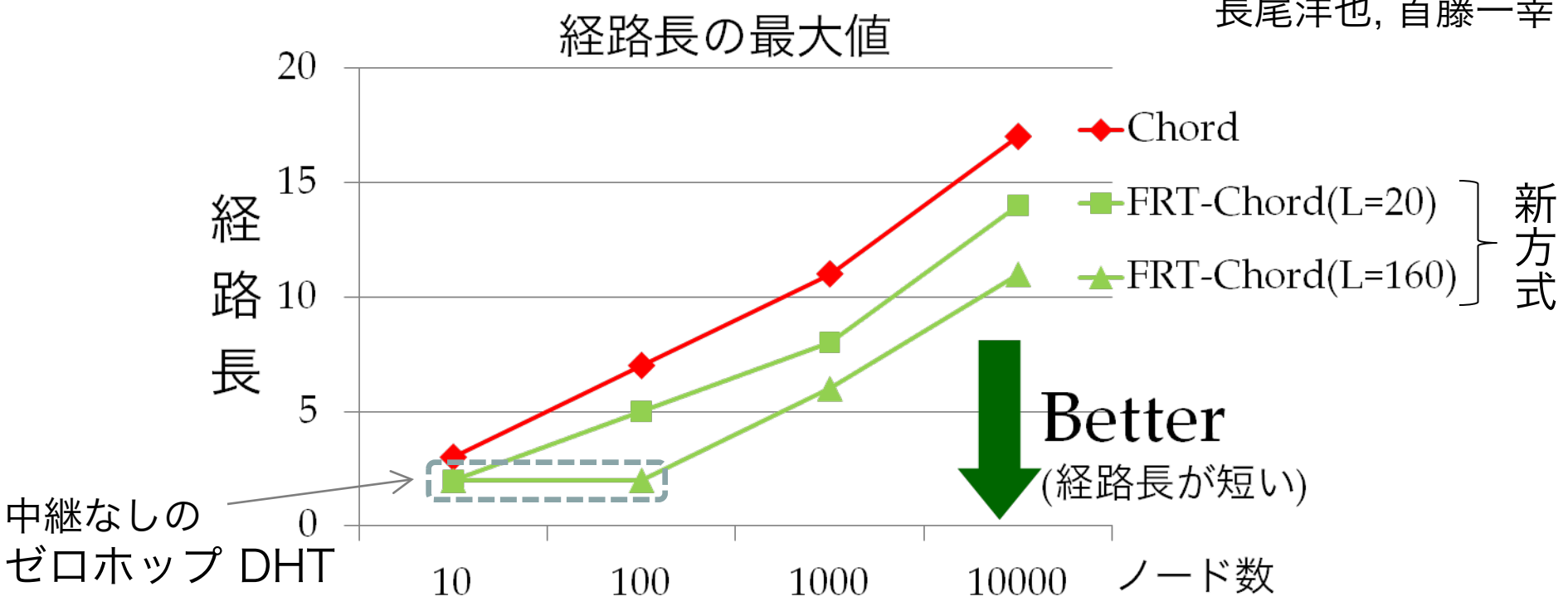
40ヶ国 669台 で DHT を構成

PLANETLAB
An open platform for developing, deploying, and accessing planetary-scale services

両方のいいとこ取り

・新しいルーティング アルゴリズムを開発

長尾洋也, 首藤一幸



- ノード数 少 → 担当サーバに直接到達
- ノード数 多 → 中継を許容しきちんと到達

高性能 両立

高 スケーラビリティ

今後

- NoSQL / key-value ストアへの実装
  - Cassandra？ 楽天 ROMA？
- 「柔軟な経路表管理」概念の応用
  - よりよい DHT
    - ネットワーク近接性の考慮
    - 複数の表の管理
  - 経路表爆発の起きない広域ルーティング方式
  - …

内容

- key-value ストア
  - ネットワーク越しに使う連想配列, map
- NoSQL データストア
  - RDB ではないデータストア
  - ムーブメント
- 特徴とその根源
  - キー, 行を単位として分散
  - 分散トランザクションを避ける
- さらなるスケーラビリティを
  - peer-to-peer 由来の新方式