HITACHI
Inspire the Next

分散Key-Valueストア「okuyama」&「Riak」書込み性能検証

2014/7/5

株式会社 日立ソリューションズ
オープンソース技術開発センタ

倉又 裕輔

自己紹介

HITACHI
Inspire the Next

■ 名前：

倉又 裕輔（くらまた ゆうすけ）

■ 所属：

（株）日立ソリューションズ　オープンソース技術開発センタ

■ 担当業務：

OSSのNoSQL技術の調査、検証

・エンタープライズ利用に向けた調査と検証

・技術情報の社内外への発信

HITACHI
Inspire the Next

発表の流れ

1. 検証の背景
2. ベンチマーク方法
3. ベンチマーク結果
4. 考察

HITACHI
Inspire the Next

1. 検証の背景

NoSQL、Key-Value ストアに着目した背景

なぜ「okuyama」と「Riak」を検証したのか

1.1 IT技術動向

HITACHI
Inspire the Next

近年のビッグデータへの期待から、IoTへの関心が高まっている

2017年までに20%以上の企業が、
従来の情報機器とは異なる**センサや組み込み機器**などを
利用し、ビジネスを取り巻く環境をデジタル化する。
多様な情報を収集、分析することで
製品やサービスをより効果的に提供するために、
モノのインターネットに関する新たな取り組みを行う。

☆引用 ： 2014年4月17日 ガートナー ジャパン株式会社 プレスリリース
【 http://www.gartner.co.jp/press/html/pr20140417-01.html 】

1.2. IoT(センサー)に適するデータストア要件

HITACHI
Inspire the Next

センサーデータの特徴

1）非構造データ

2）記録回数/時間： 多

3）センサー数： 大量

4）データ送信頻度： 高

データストアの要件

A）非構造データの保存

B）大量データの保存

C）高速な書込み性能

1.3. NoSQL適合の課題

HITACHI
Inspire the Next

NoSQL(Not only SQL)

リレーショナルデータベース(RDB)以外のデータストアの総称
SQLを使用せずにデータを操作する

☆得意

- 非構造データの保存(スキーマレス)
- スケールアウト(大量データの保存)
- 高速な書込み、参照処理

★苦手

- トランザクション処理
- 複雑な検索、集計処理

データストアの要件に適合

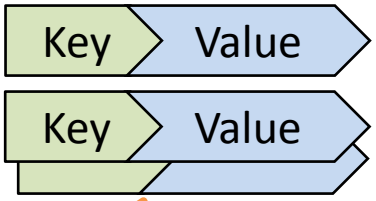
しかし、性能に関する情報が不足している
特に書込み性能に注目し
知見を増やすために検証を実施した

1.4. 検証対象NoSQLの検討(1/2)

HITACHI
Inspire the Next

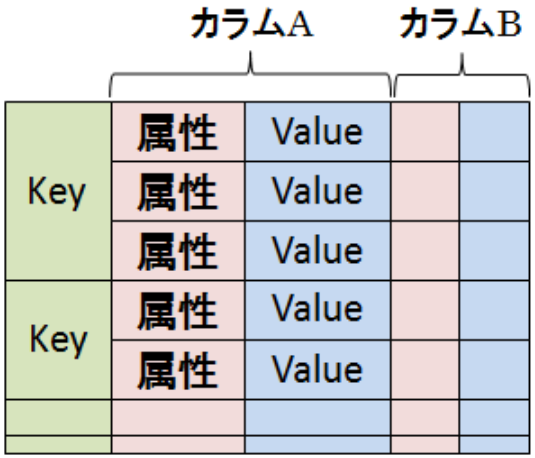
NoSQLの分類(データモデル)

Key-Value ストア(KVS)
- okuyama
- Riak
- memcached
- Redis　　など

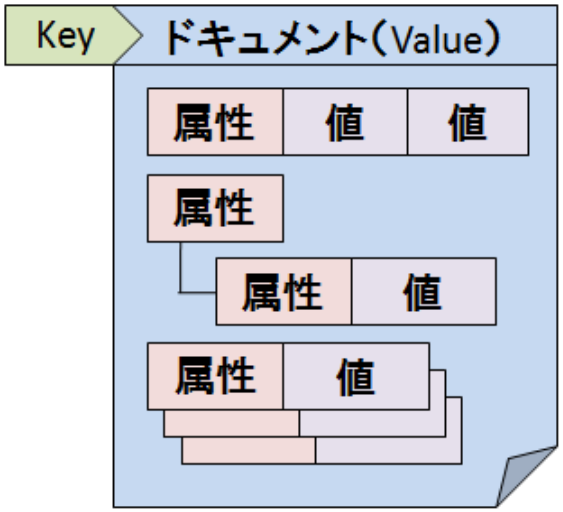


データが
シンプル

カラム指向
- Cassandra
- Hbase
- Hypertable　　など



ドキュメント指向
- MongoDB
- CouchDB
- Couchbase Server　　など



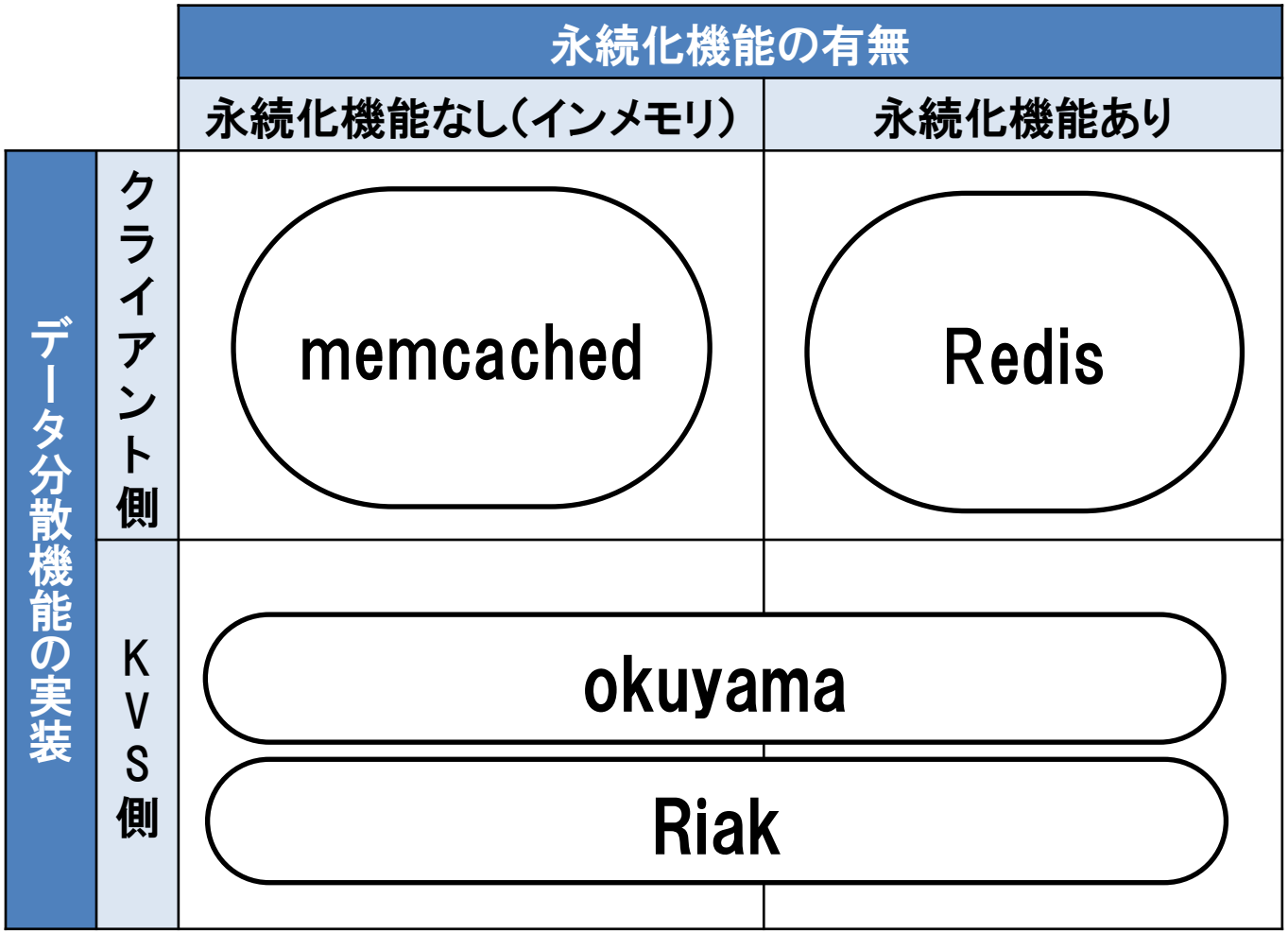
KVSは小さい多数データの蓄積、高速なデータ読み書きが得意

センサーデータに適している

1.5. 検証対象NoSQLの検討(2/2)

HITACHI
Inspire the Next

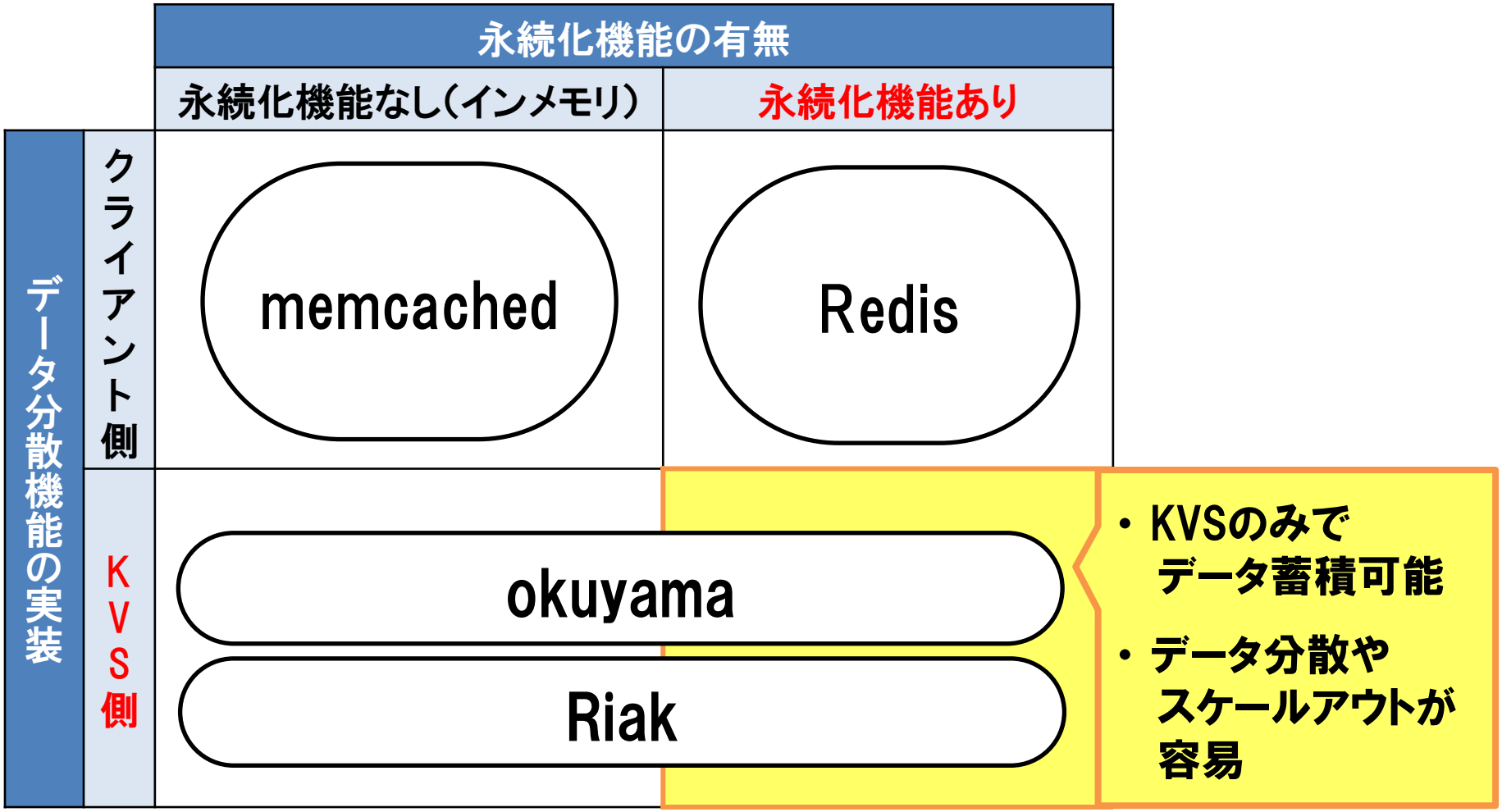
KVSの分類



1.5. 検証対象NoSQLの検討(2/2)

HITACHI
Inspire the Next

KVSの分類

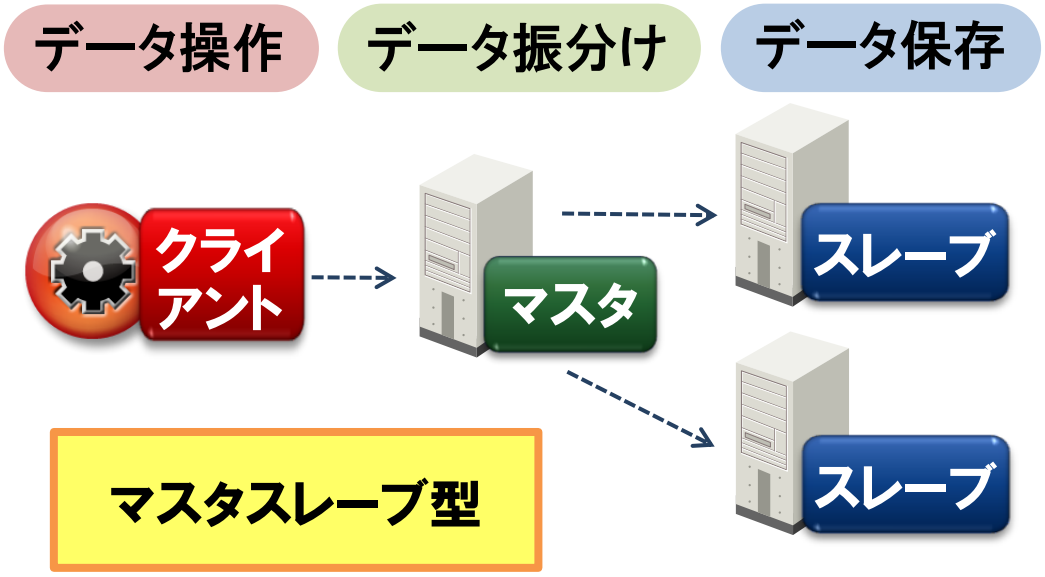


「okuyama」と「Riak」を検証対象として選択

1.6. okuyama の特徴

HITACHI
Inspire the Next

開発元：神戸デジタル・ラボ（ http://okuyama-project.com/ja/index.html ）



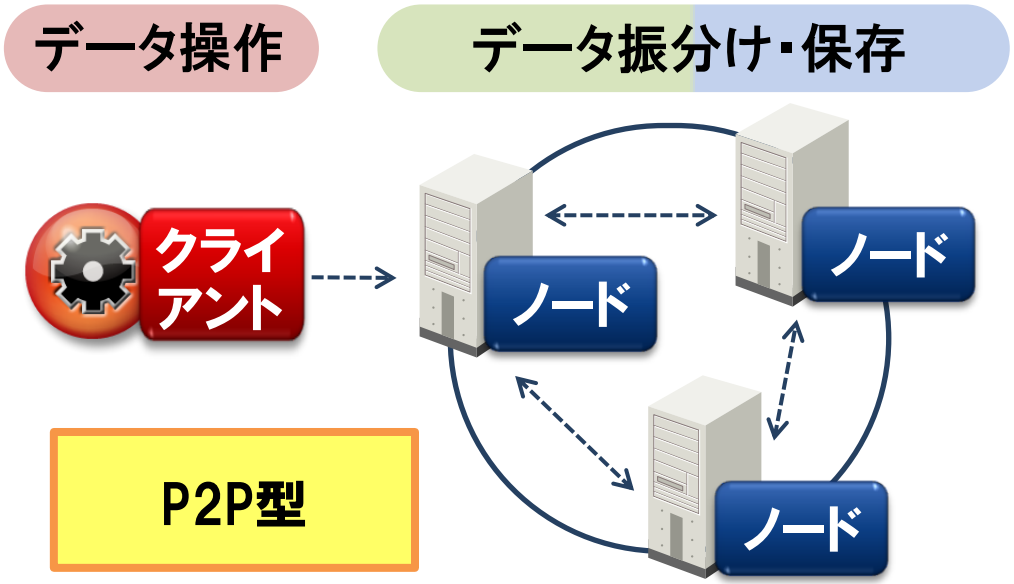
- 冗長化により
  単一障害点の無い
  クラスタを構築可能
- 無停止でスケールアウト可能
- データ保存先を選択可能
  （ディスク・メモリ・併用）

okuyama	
開発元	株式会社　神戸デジタル・ラボ
ライセンス	GPL v3（商用ライセンス・サポートあり）
開発言語	Java
分散アーキテクチャ	マスタ-スレーブ型
ユースケース	ECサイト、検索エンジン、キャッシュサーバ、ログ管理、など

1.7. Riak の特徴

HITACHI
Inspire the Next

開発元：Basho Technologies (http://basho.co.jp/riak/)



- 単一障害点なし
- 無停止でスケールアウト可能
- データ保存先を選択可能
  （ディスク・メモリ・併用）

Riak	
開発元	Basho Technologies, Inc.
ライセンス	Apache License, Version 2.0（商用ライセンス・サポートあり）
開発言語	Erlang/C
分散アーキテクチャ	P2P型
ユースケース	ECサイト、キャッシュサーバ、オブジェクトストレージ、など

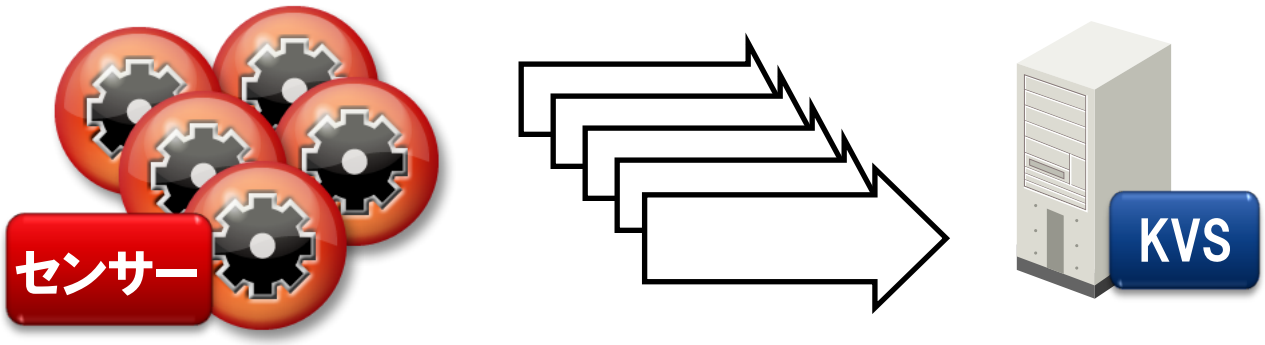
HITACHI
Inspire the Next

2. ベンチマーク方法

検証の目的、観点

検証環境、ベンチマークツール
ベンチマークの内容

2.1. 検証の目的



多数のセンサーが
高頻度に大量件数のデータを同時送信

同時書込み性能が重要

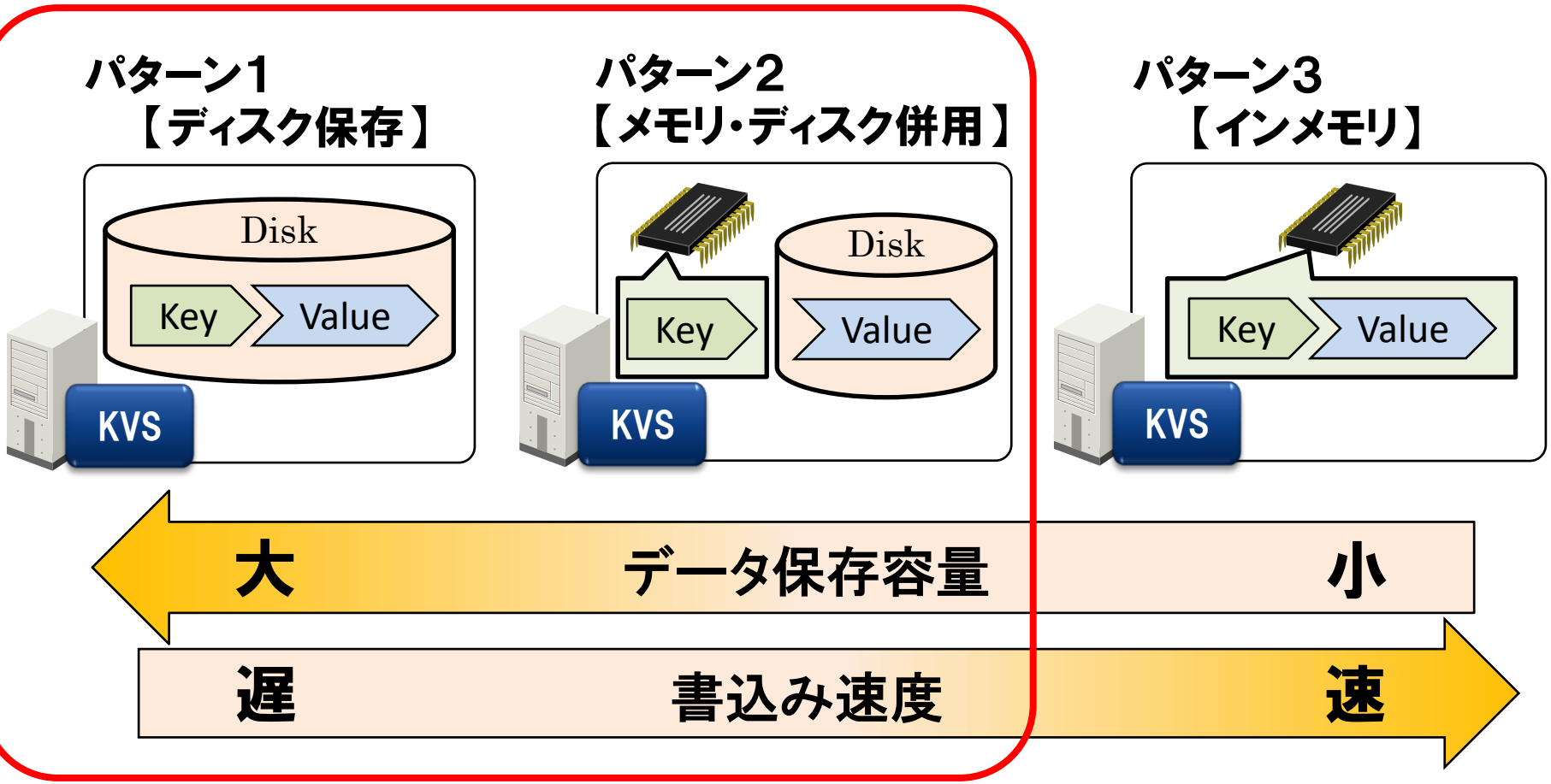
実際どのくらい速いのか？ベンチマーク実施

スループット（書込み数/秒）と**レイテンシ**（書込み時間）を計測

2.2. 検証パターン

HITACHI
Inspire the Next

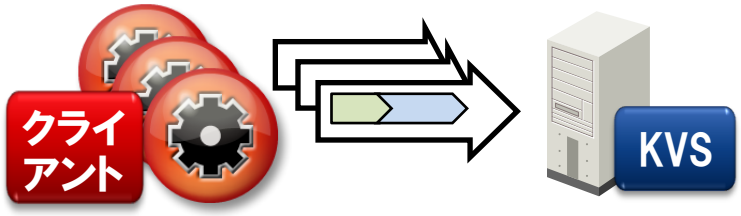
「okuyama」と「Riak」の特性は構成によって異なる



大量データ保存が要件のため、パターン1と2を検証

2.3. 検証の観点

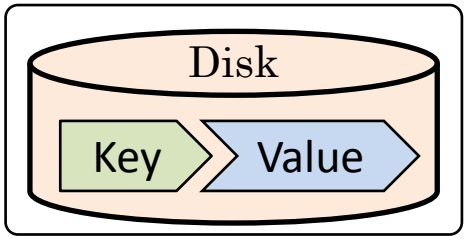
観点1：同時書込み性能



同時アクセス数を増やして、
性能変化を検証

観点2：各検証パターンごとの懸念点を検証

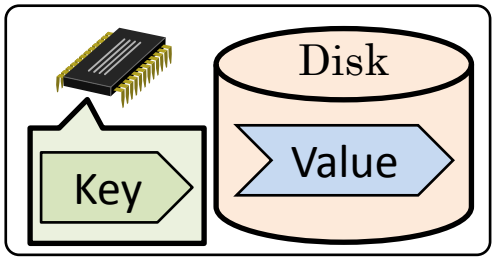
【ディスク保存】



書込み速度が遅い？

スケールアウトによる
性能向上を検証

【メモリ・ディスク併用】



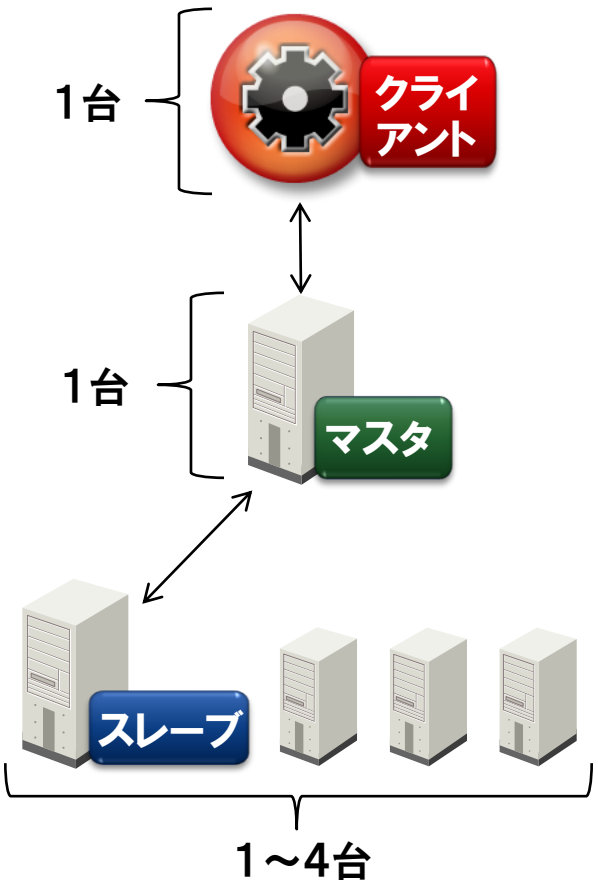
大量データ保存によるメモリ消費時の性能？

メモリが無くなるまで書込み、
性能変化を検証

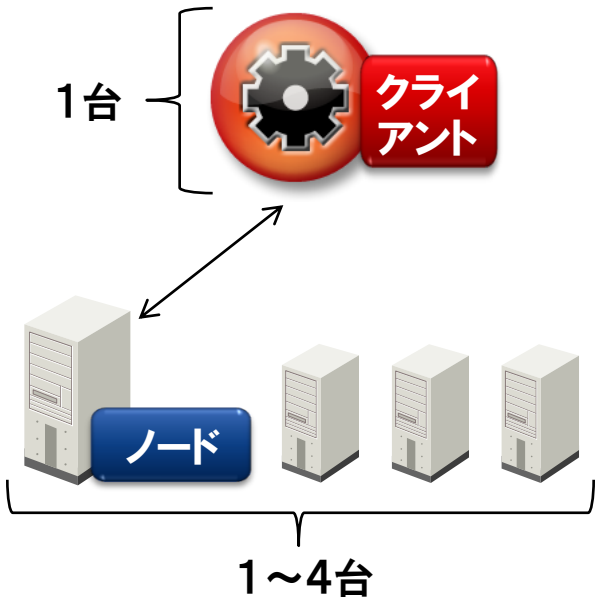
2.4. 検証環境

HITACHI
Inspire the Next

okuyama v0.9.5（1ノード/台）



Riak v1.4.6（1ノード/台）



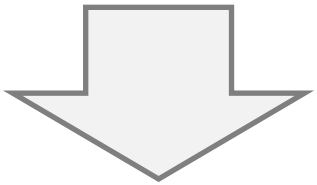
CPU	Intel(R) Core(TM) i5 CPU 520
メモリ	4GB
HDD(SATA)	750GB, 5400RPM
OS	CentOS 6.5 64bit

※データの複製・レプリケーション無し

2.5. ベンチマークツール

HITACHI
Inspire the Next

他ベンチマークと比較できるよう、
一般に利用実績のあるベンチマークツールを使いたい



NoSQL用ベンチマークツール

「Yahoo! Cloud Serving Benchmark（YCSB）」を利用

Yahoo! Cloud Serving Benchmark（YCSB）	
開発元	Yahoo! Inc.
ライセンス	Apache License, Version 2.0
開発言語	Java
使用バージョン	0.1.4

参考：https://github.com/brianfrankcooper/YCSB/wiki

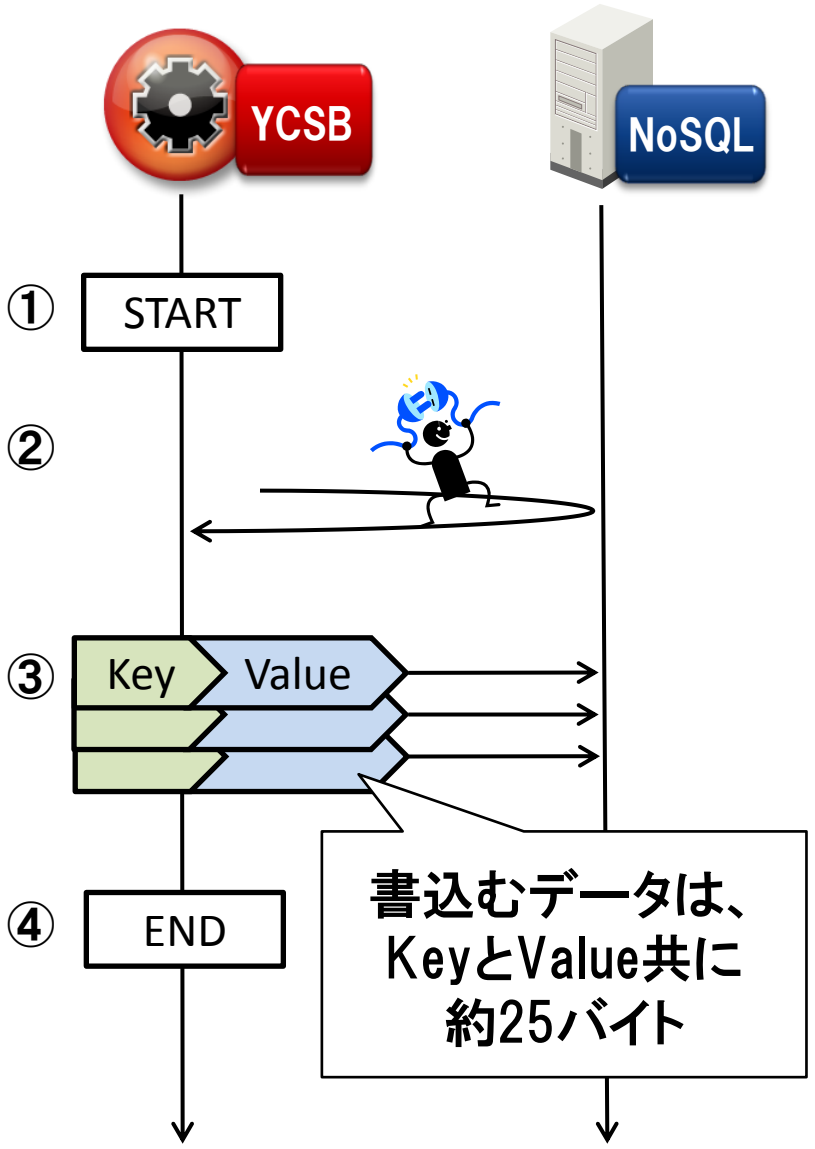
2.6. YCSBの動き

①同時アクセス数(スレッド数)、
　書込むデータ件数を指定し実行

②スレッドごとにNoSQLへ接続

③書込み処理開始
　全件を処理するまで繰り返し
　1件単位のスループットと
　レイテンシを計測

④全件の処理を終えたら終了
　スループット・レイテンシの
　計測結果を集計し出力



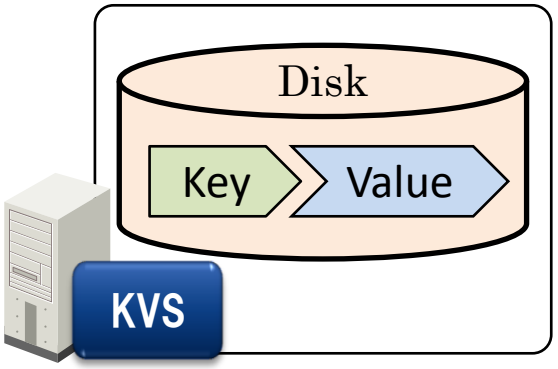
HITACHI
Inspire the Next

3. ベンチマーク結果

3.1. 検証パターンと観点の確認

HITACHI
Inspire the Next

【ディスク保存】



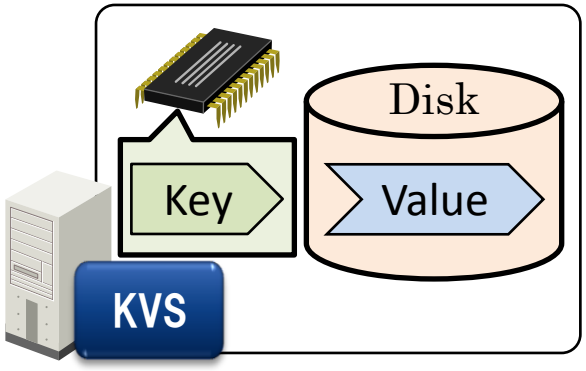
観点1：同時書込み性能

観点2：スケールアウト性能

okuyama　検証①

Riak　検証②

【メモリ・ディスク併用】



観点1：同時書込み性能

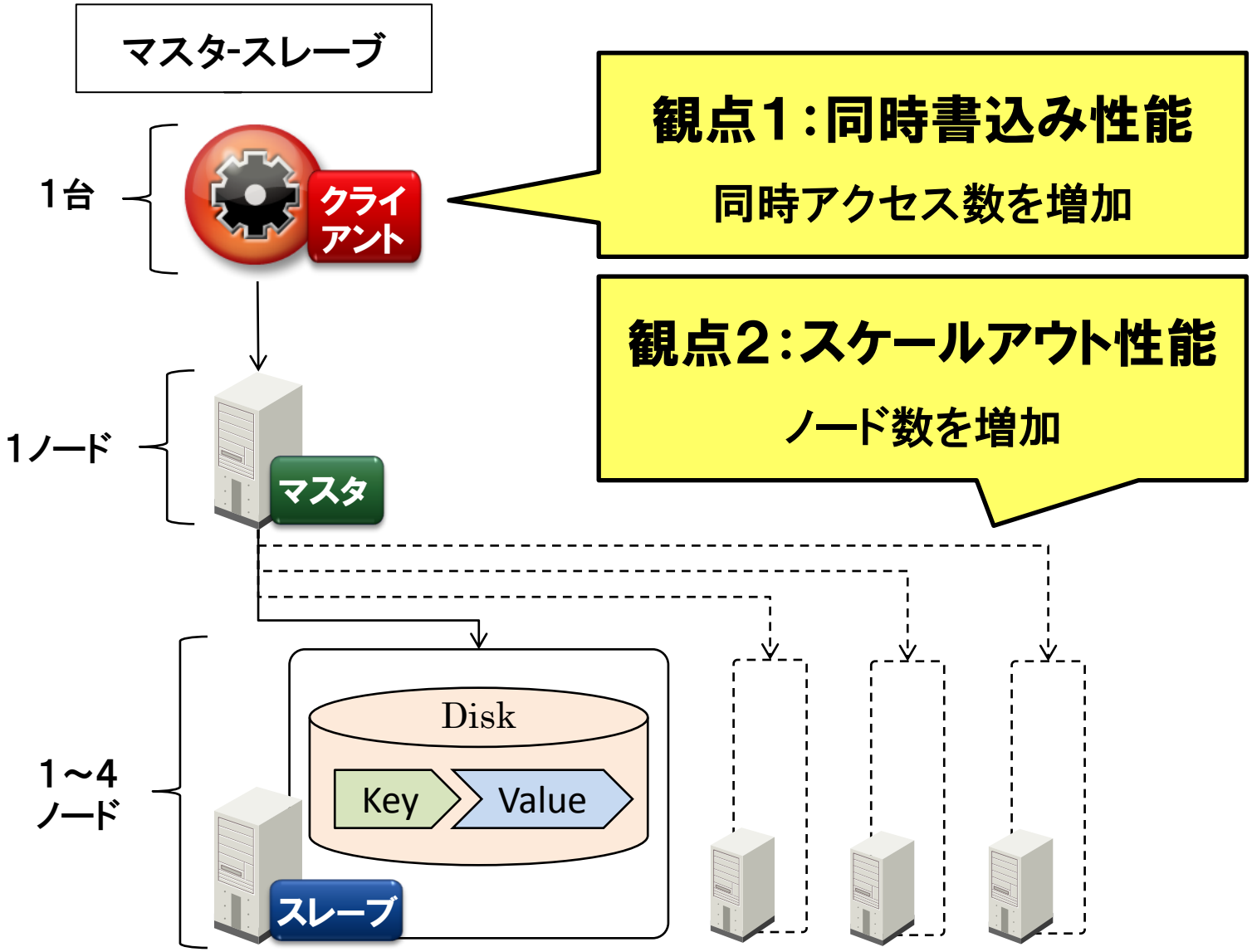
観点2：メモリ消費時の性能

okuyama　検証③

Riak　検証④

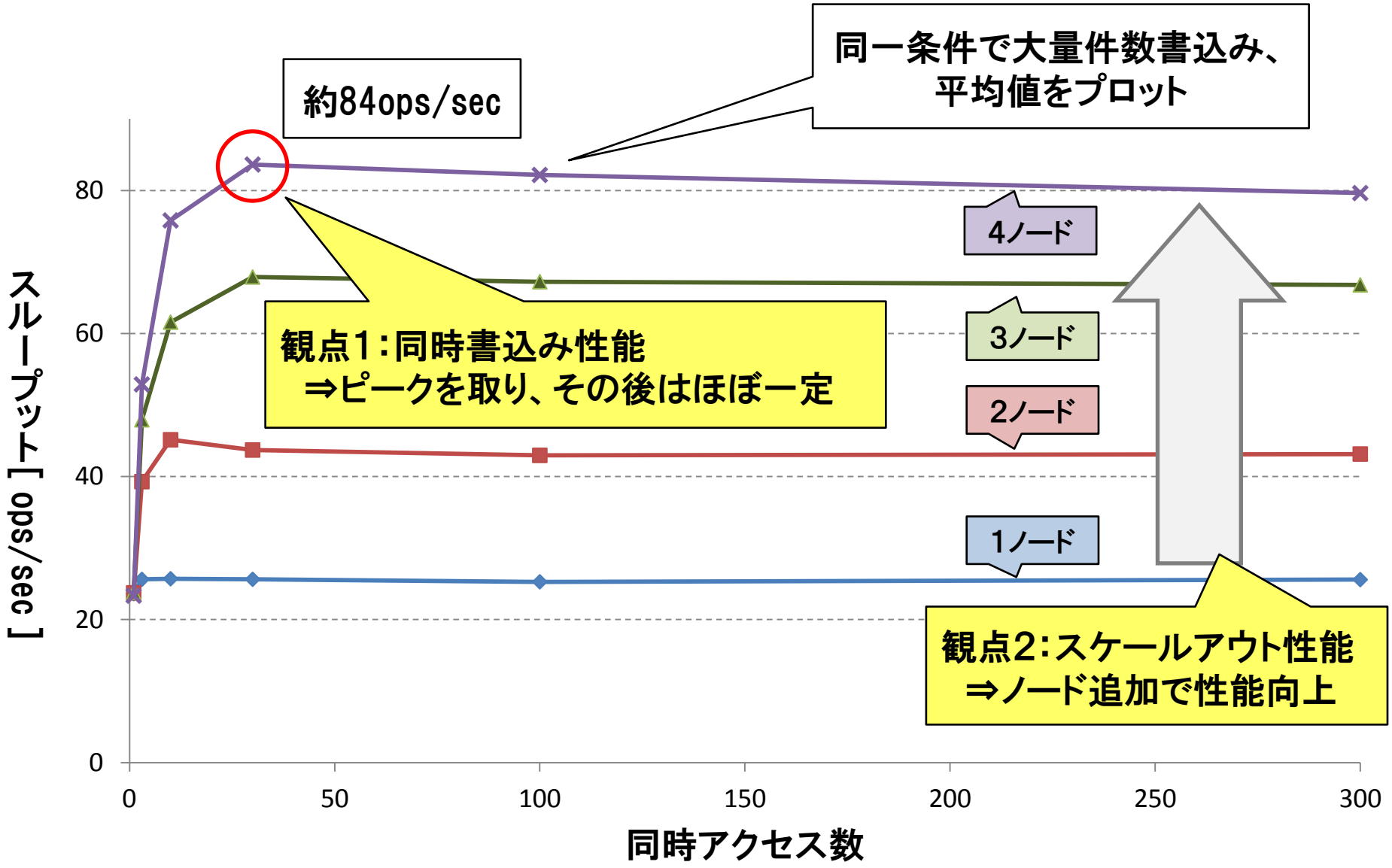
4つの検証ごとにそれぞれ説明

3.2. 検証① okuyama -【ディスク保存】



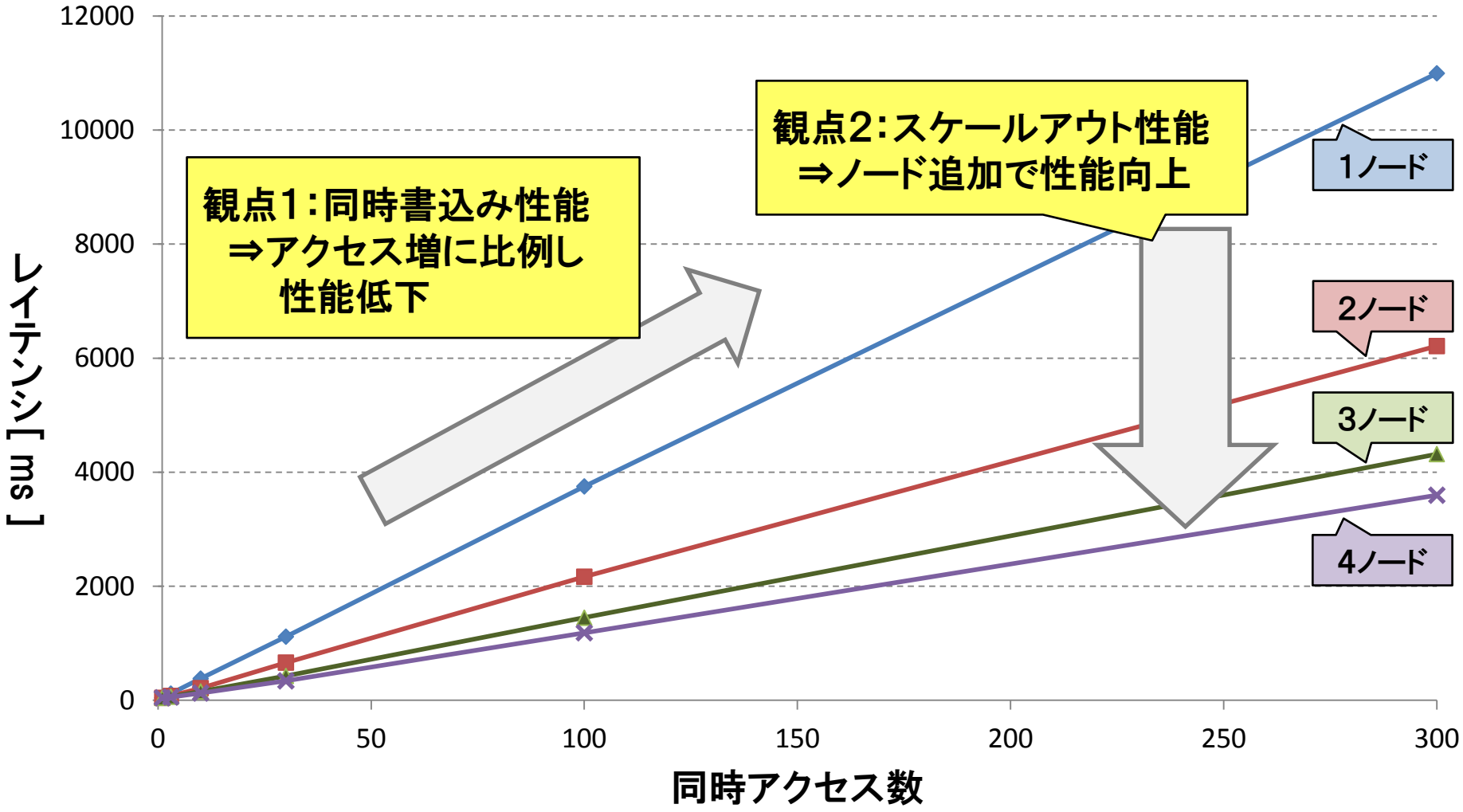
3.2. 検証① okuyama -【ディスク保存】

HITACHI
Inspire the Next



3.2. 検証① okuyama -【ディスク保存】

HITACHI
Inspire the Next



3.2. 検証① okuyama -【ディスク保存】

HITACHI
Inspire the Next

わかったこと

１）同時アクセス性能

同時アクセス数が増加しても、急な性能低下はしない。
リソース使用状況から、ディスクI/Oがボトルネック。

２）スケールアウト性能

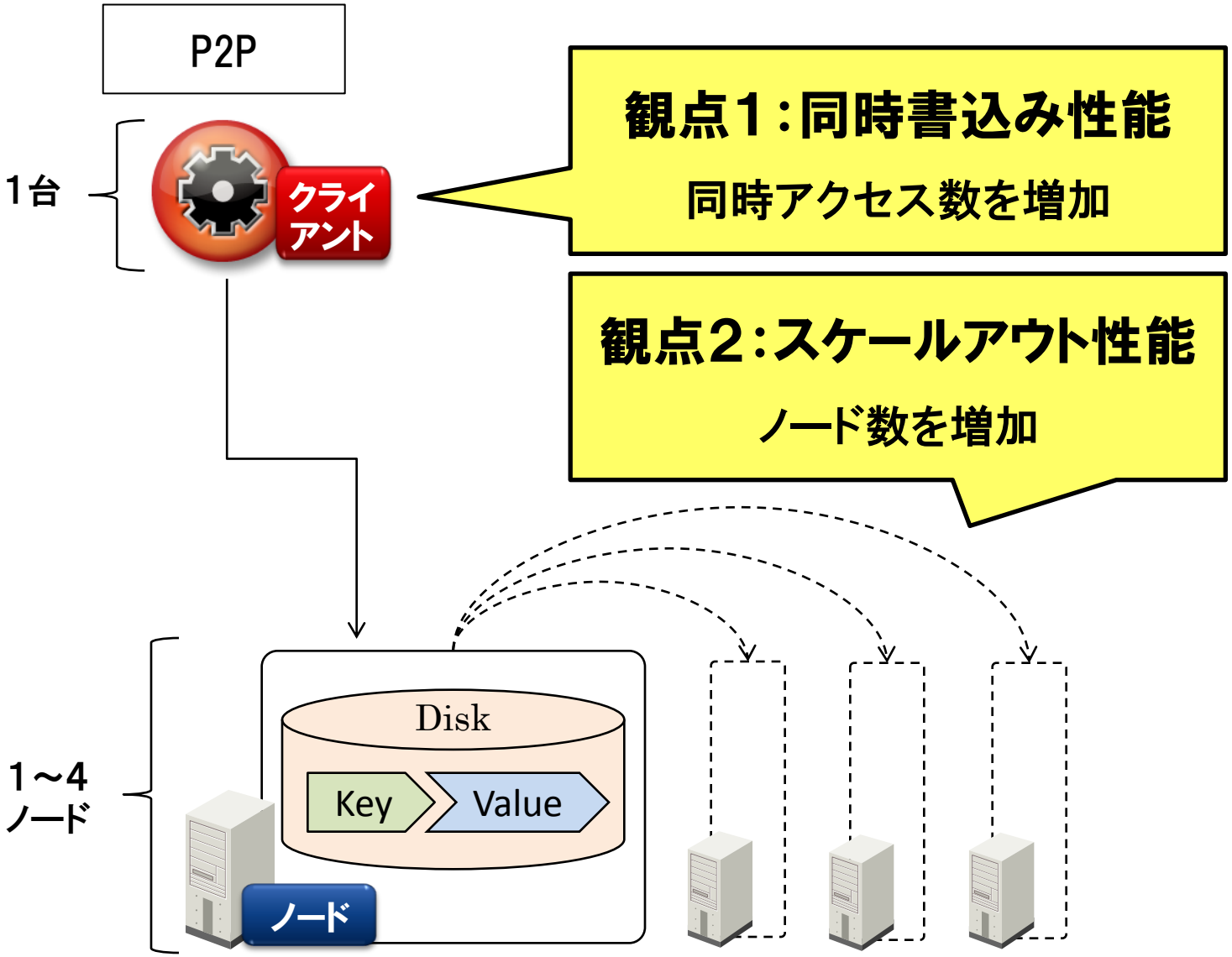
ノード追加に対して想定通り性能向上。

３）スループット

最大で約 84 ops/sec（4ノードクラスタ）

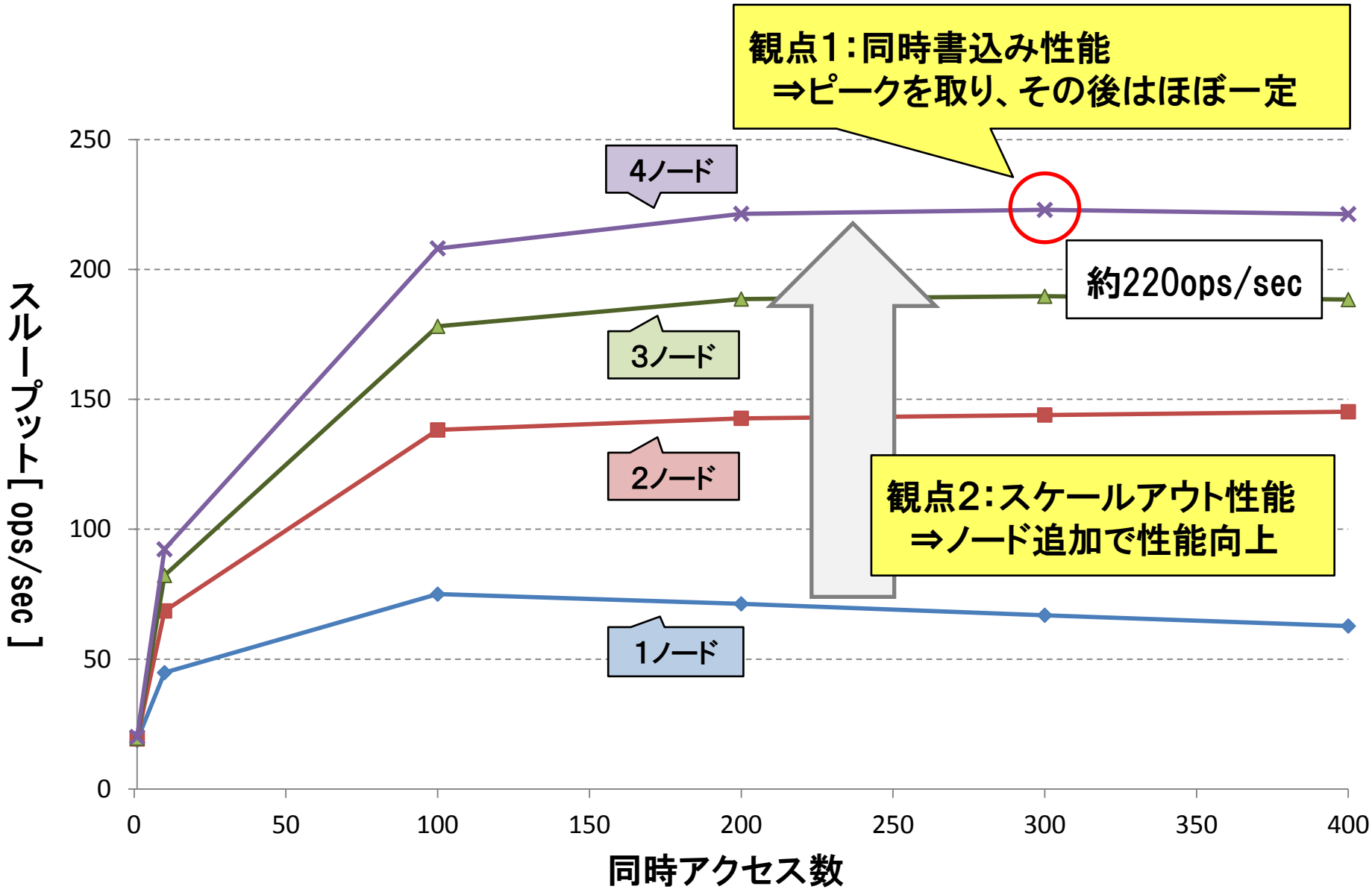
3.3. 検証② Riak -【ディスク保存】

HITACHI
Inspire the Next



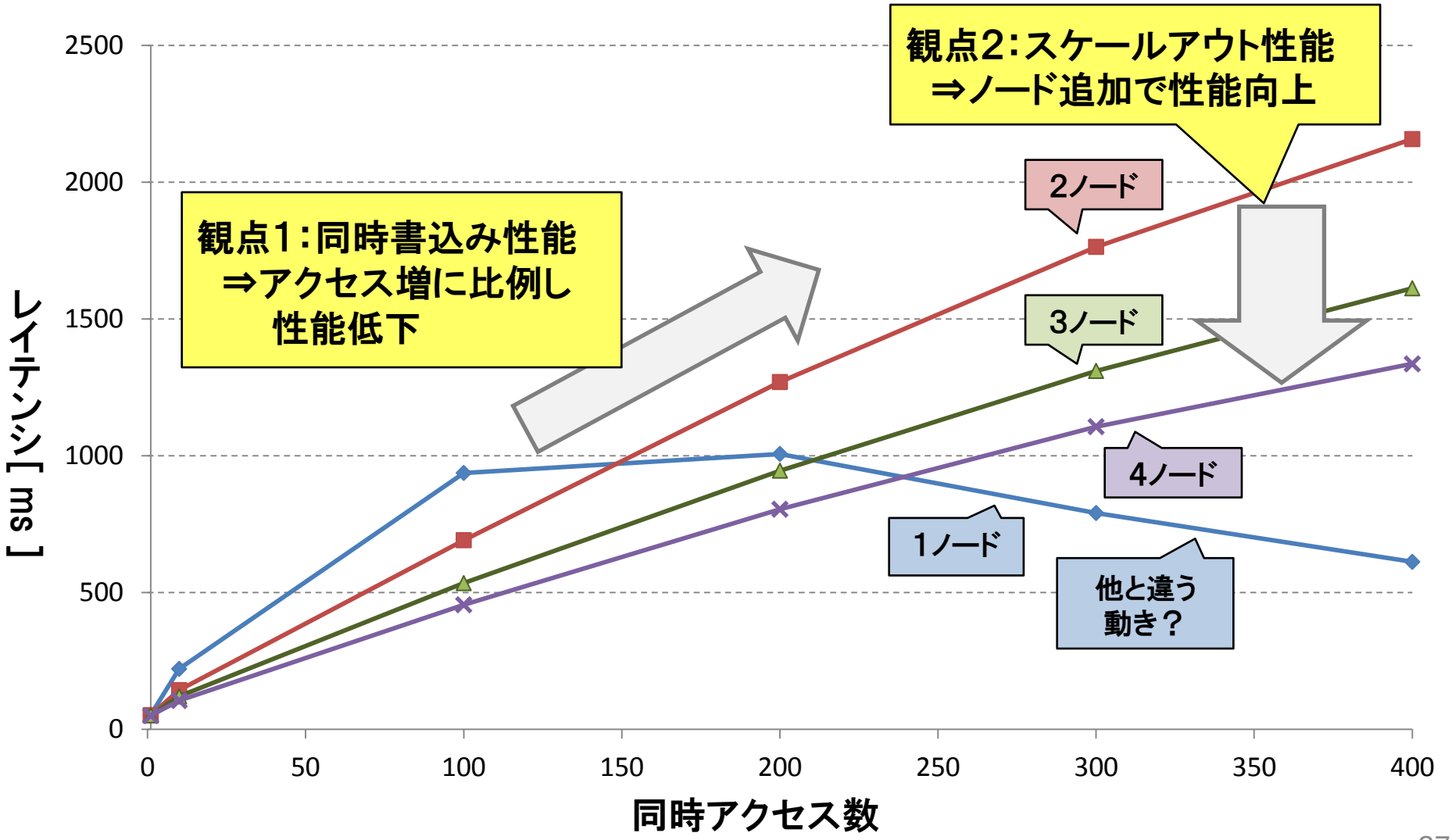
3.3. 検証② Riak -【ディスク保存】

HITACHI
Inspire the Next



3.3. 検証② Riak -【ディスク保存】

HITACHI
Inspire the Next



3.3. 検証② Riak -【ディスク保存】

HITACHI
Inspire the Next

わかったこと

１）同時アクセス性能

同時アクセス数が増加しても、急な性能低下はしない。
リソース使用状況から、ディスクI/Oがボトルネック。

２）スケールアウト特性

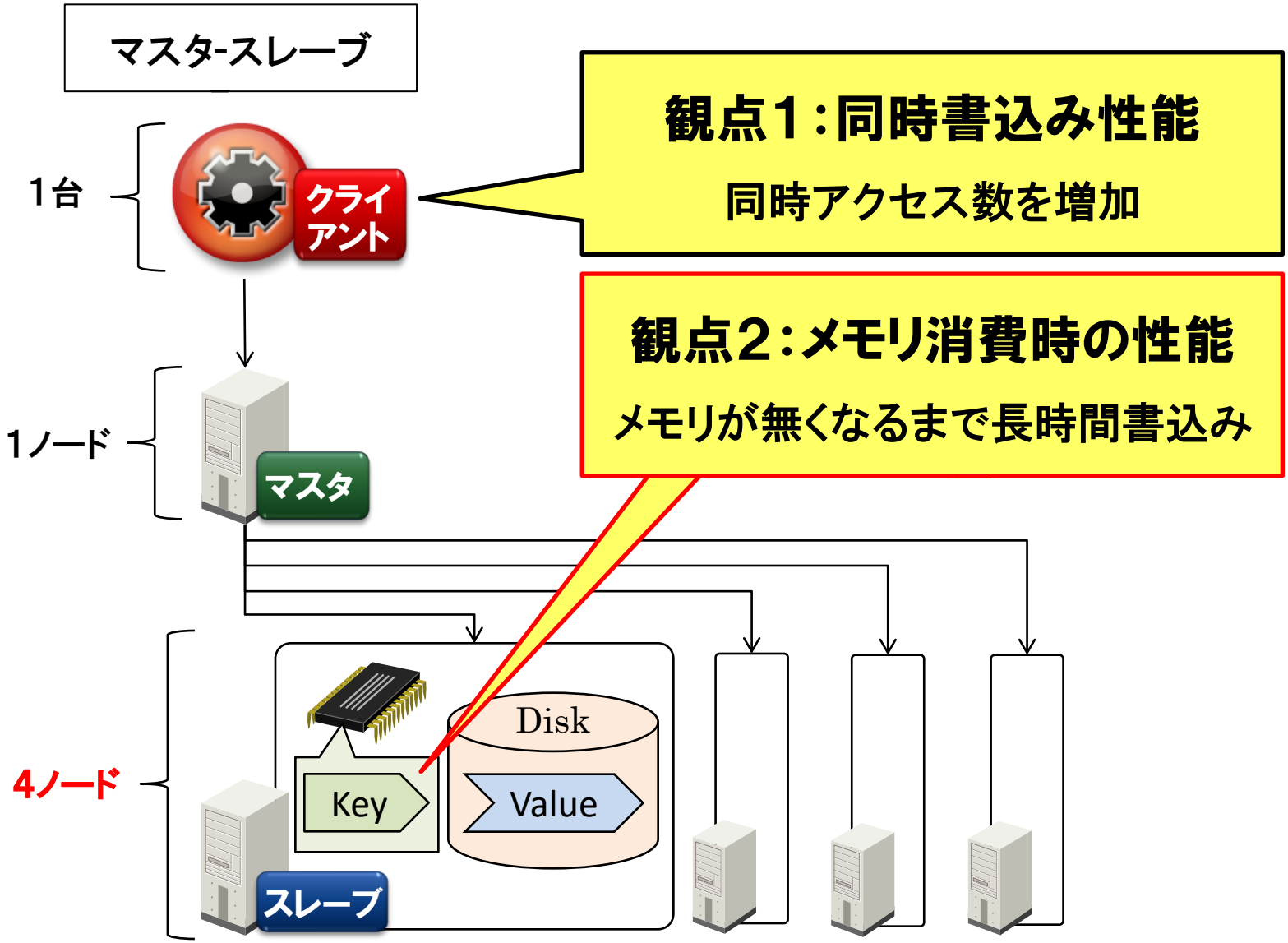
ノード追加に対して想定通り性能向上。

３）スループット

最大で約 220 ops/sec（4ノードクラスタ）

3.4. 検証③ okuyama -【メモリ・ディスク併用】

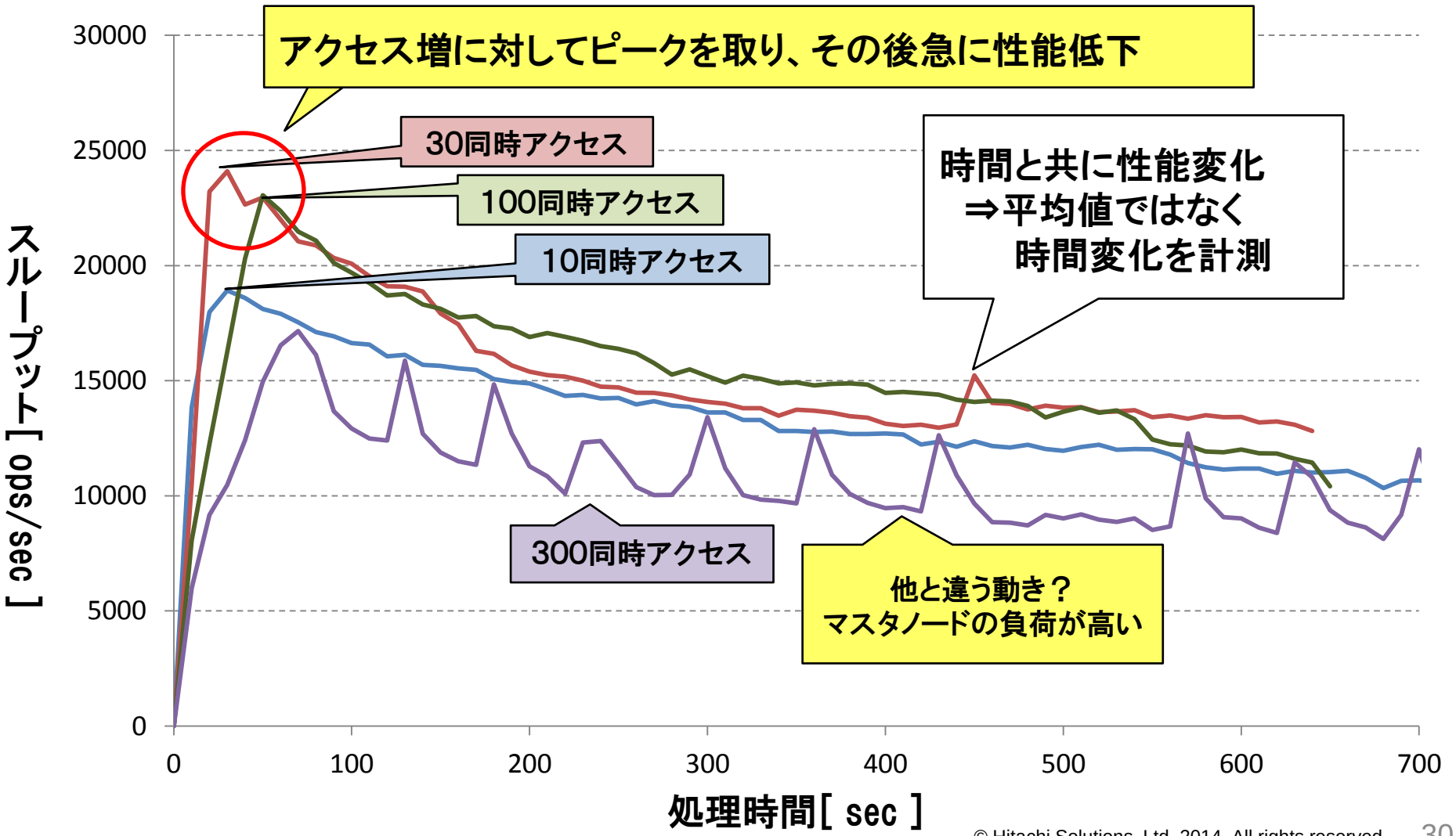
HITACHI
Inspire the Next



3.4. 検証③ okuyama –【メモリ・ディスク併用】

HITACHI
Inspire the Next

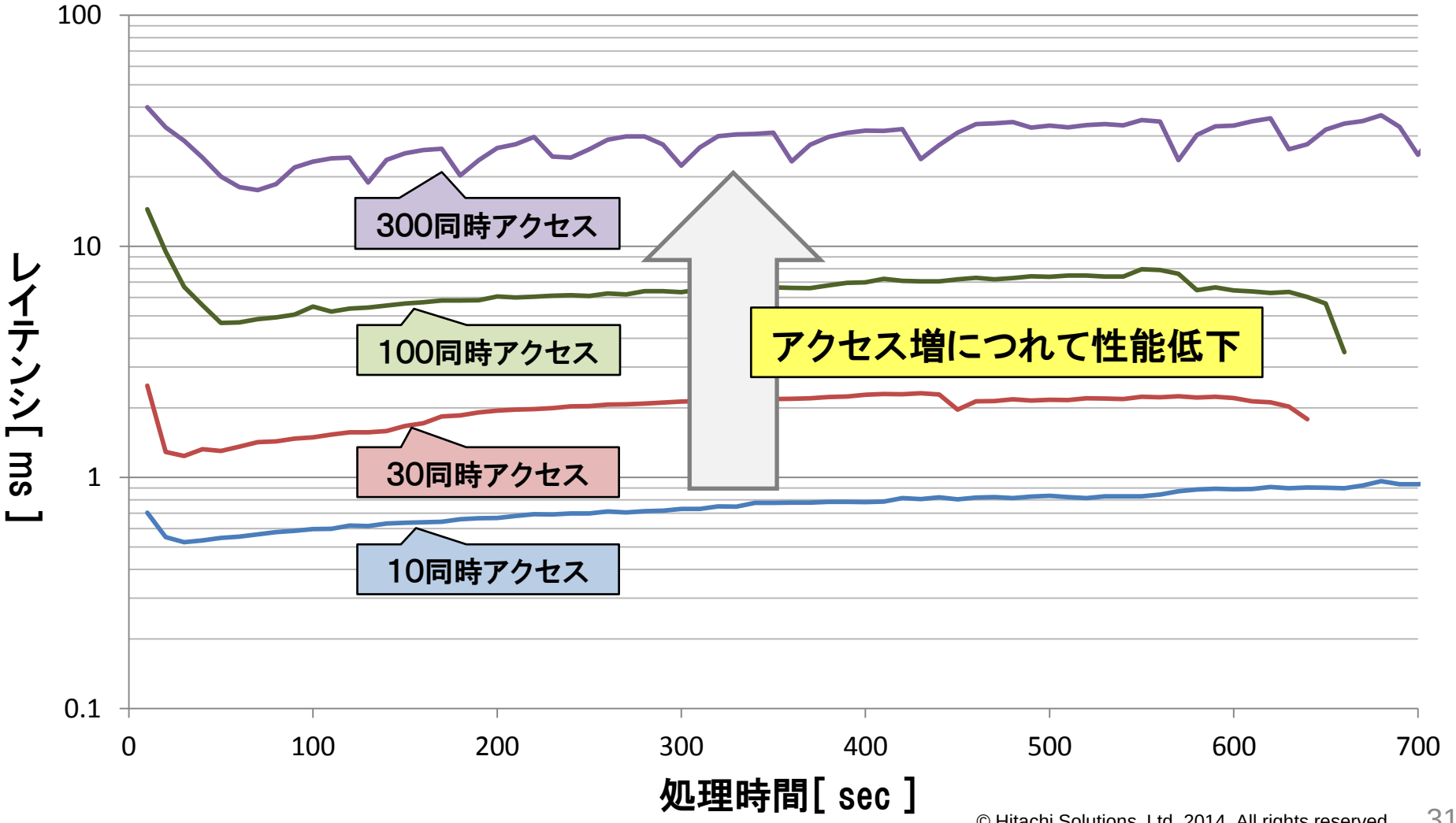
観点1：同時書込み性能：同時アクセス数を増加



3.4. 検証③ okuyama –【メモリ・ディスク併用】

HITACHI
Inspire the Next

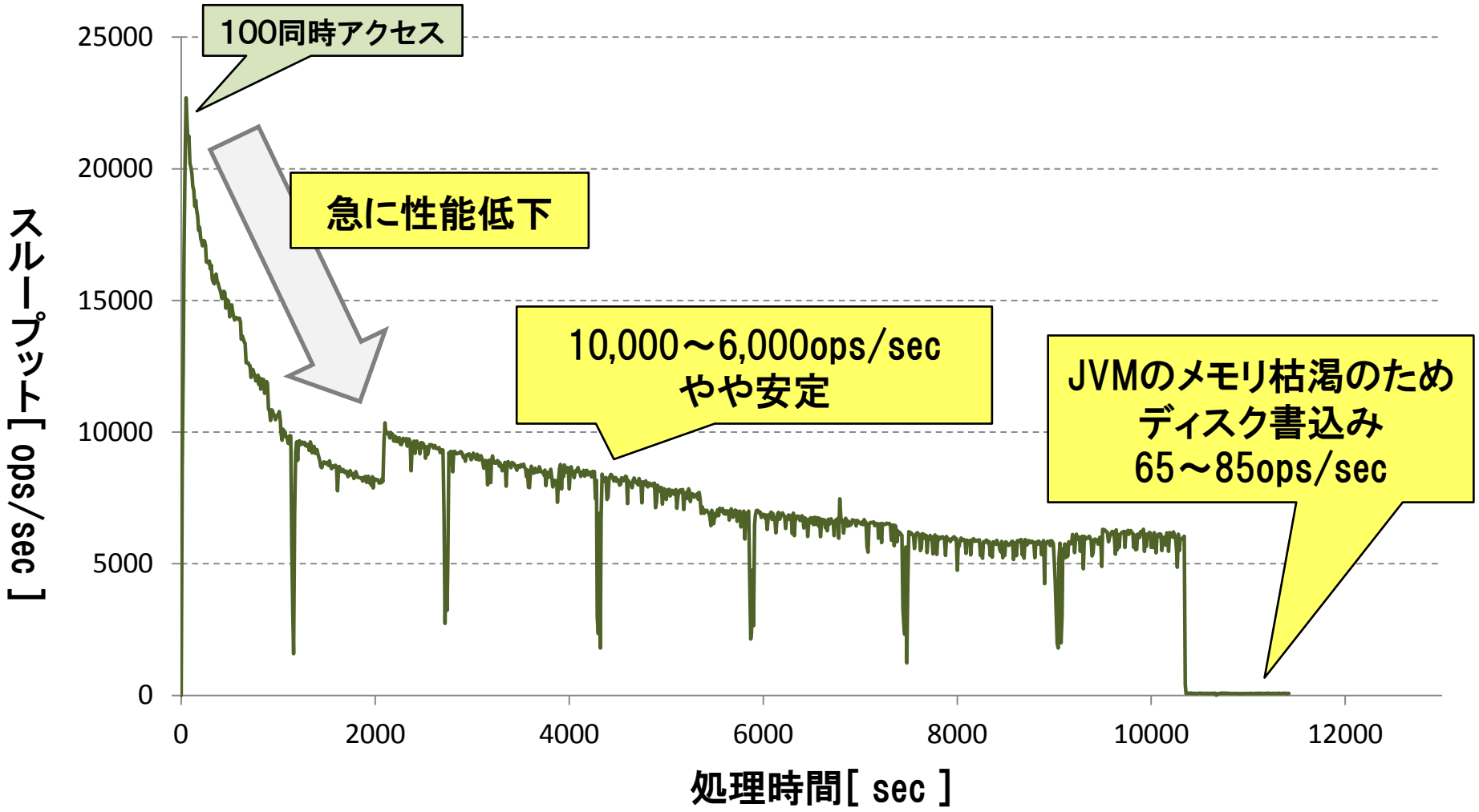
観点1：同時書込み性能：同時アクセス数を増加



3.4. 検証③ okuyama –【メモリ・ディスク併用】

HITACHI
Inspire the Next

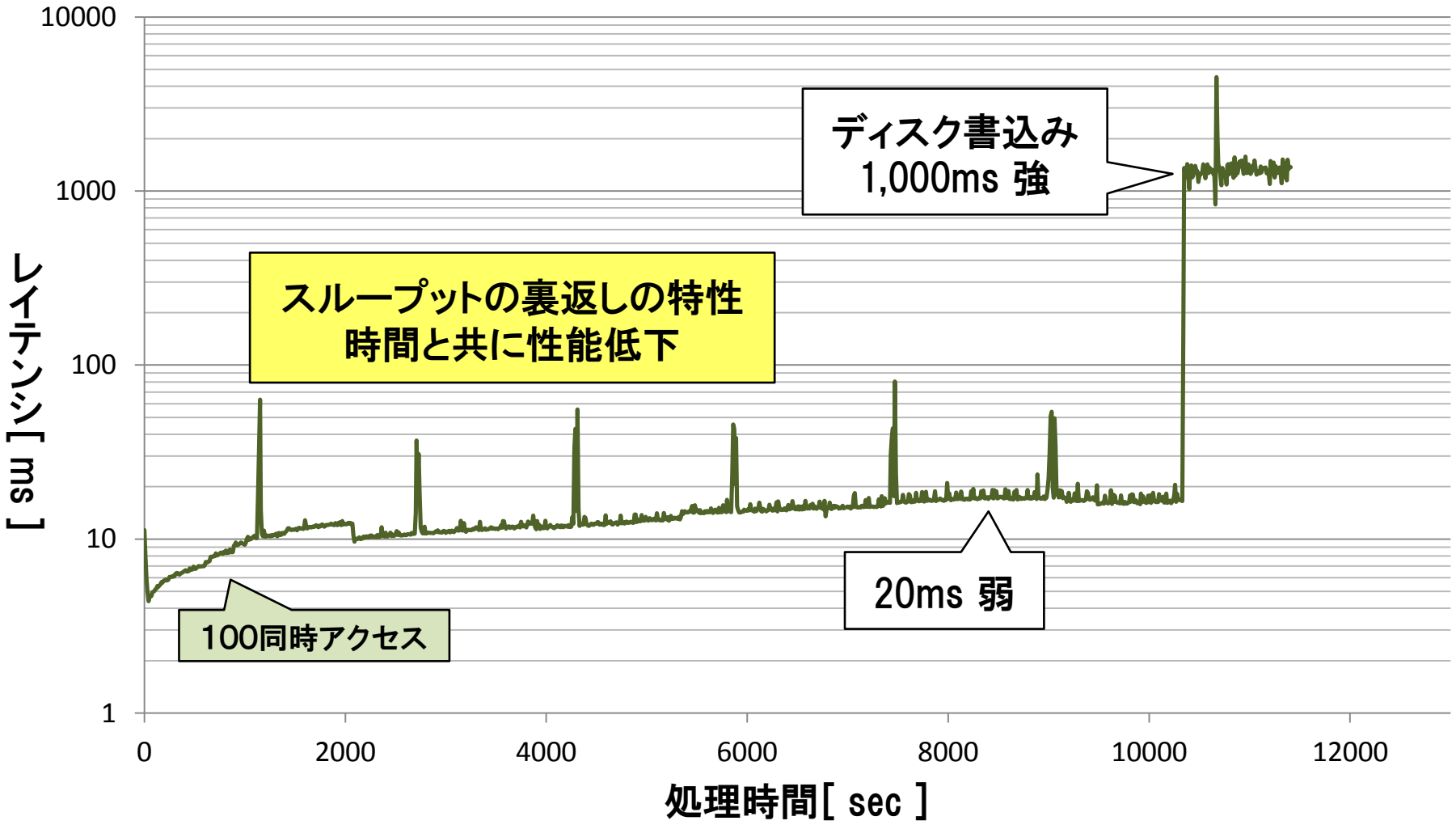
観点2：メモリ消費時の性能：メモリが無くなるまで長時間書込み



3.4. 検証③ okuyama –【メモリ・ディスク併用】

HITACHI
Inspire the Next

観点2：メモリ消費時の性能：メモリが無くなるまで長時間書込み



3.4. 検証③ okuyama -【メモリ・ディスク併用】

HITACHI
Inspire the Next

わかったこと

１）同時アクセス性能

同時アクセス数が増加すると、データ振分け処理がボトルネック。マスタノード追加による負荷分散が必要。

２）メモリ消費時の性能

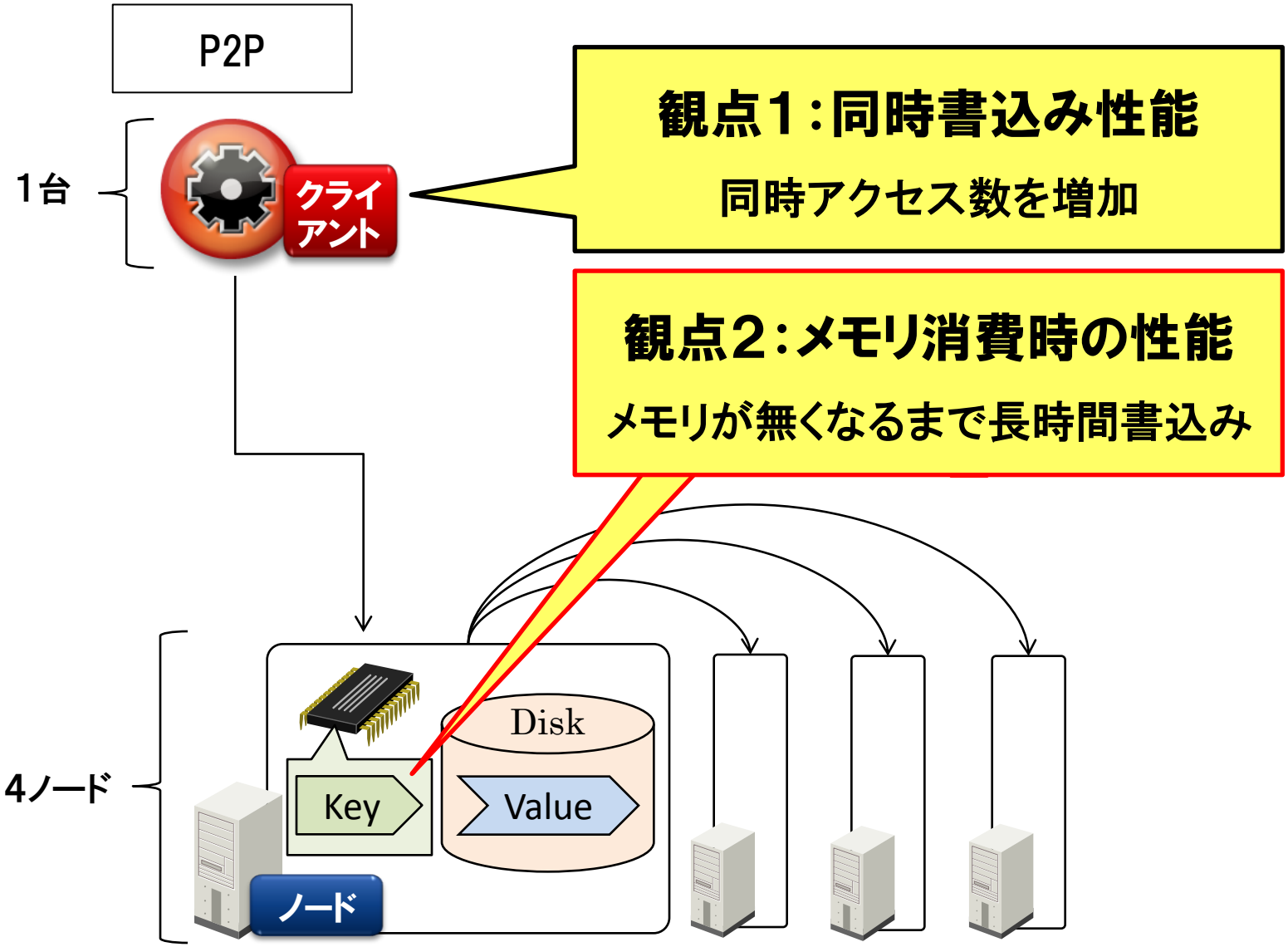
メモリ消費にともなって、徐々に性能低下。
JVMのメモリを使い切ると、ディスク書込みに切り替わり、大きく性能低下。

３）スループット（中盤の安定状態）

10,000 ops/sec ⇒ 6,000 ops/sec（4ノードクラスタ）

3.5. 検証④ Riak –【メモリ・ディスク併用】

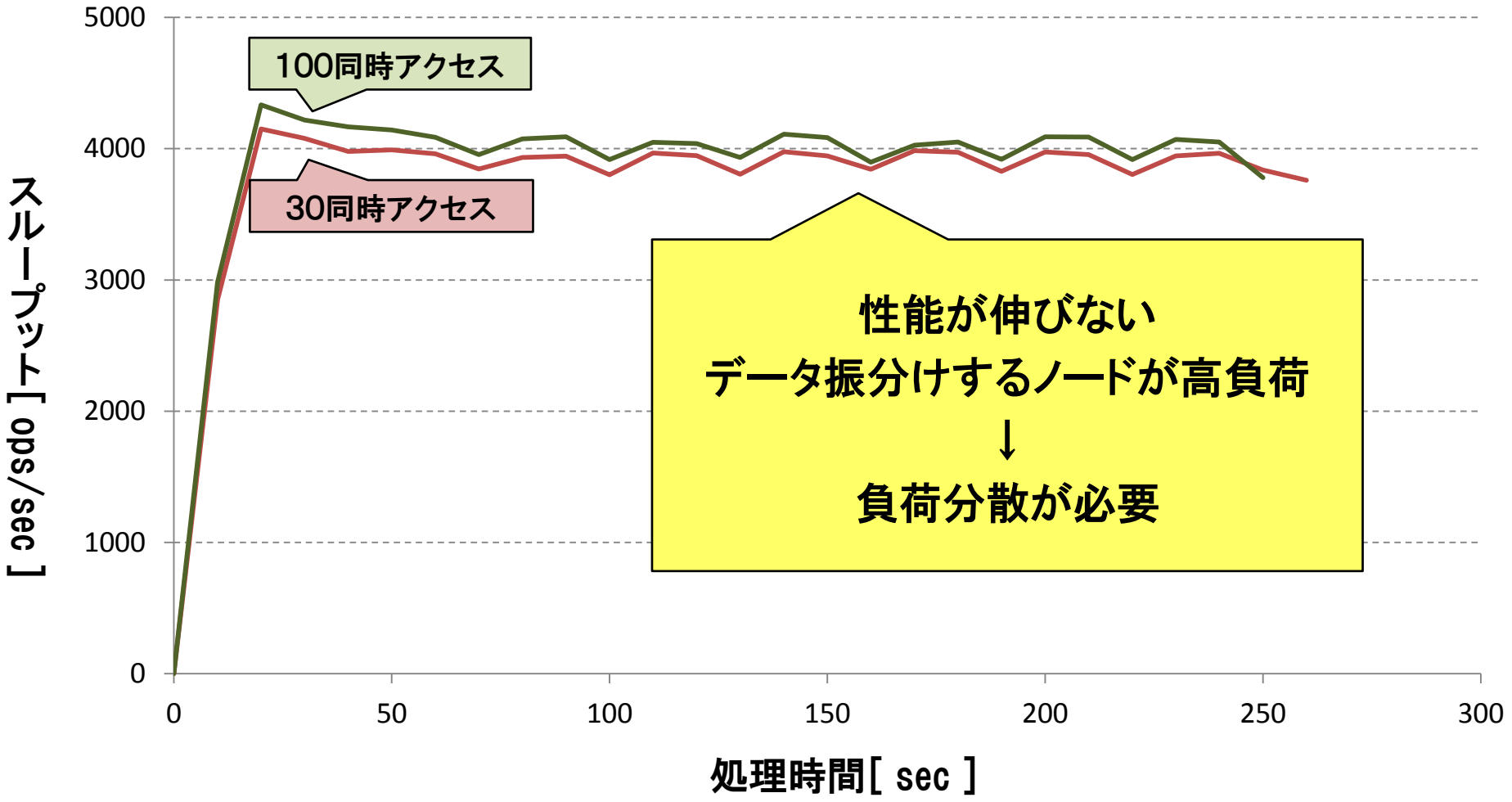
HITACHI
Inspire the Next



3.5. 検証④ Riak –【メモリ・ディスク併用】

HITACHI
Inspire the Next

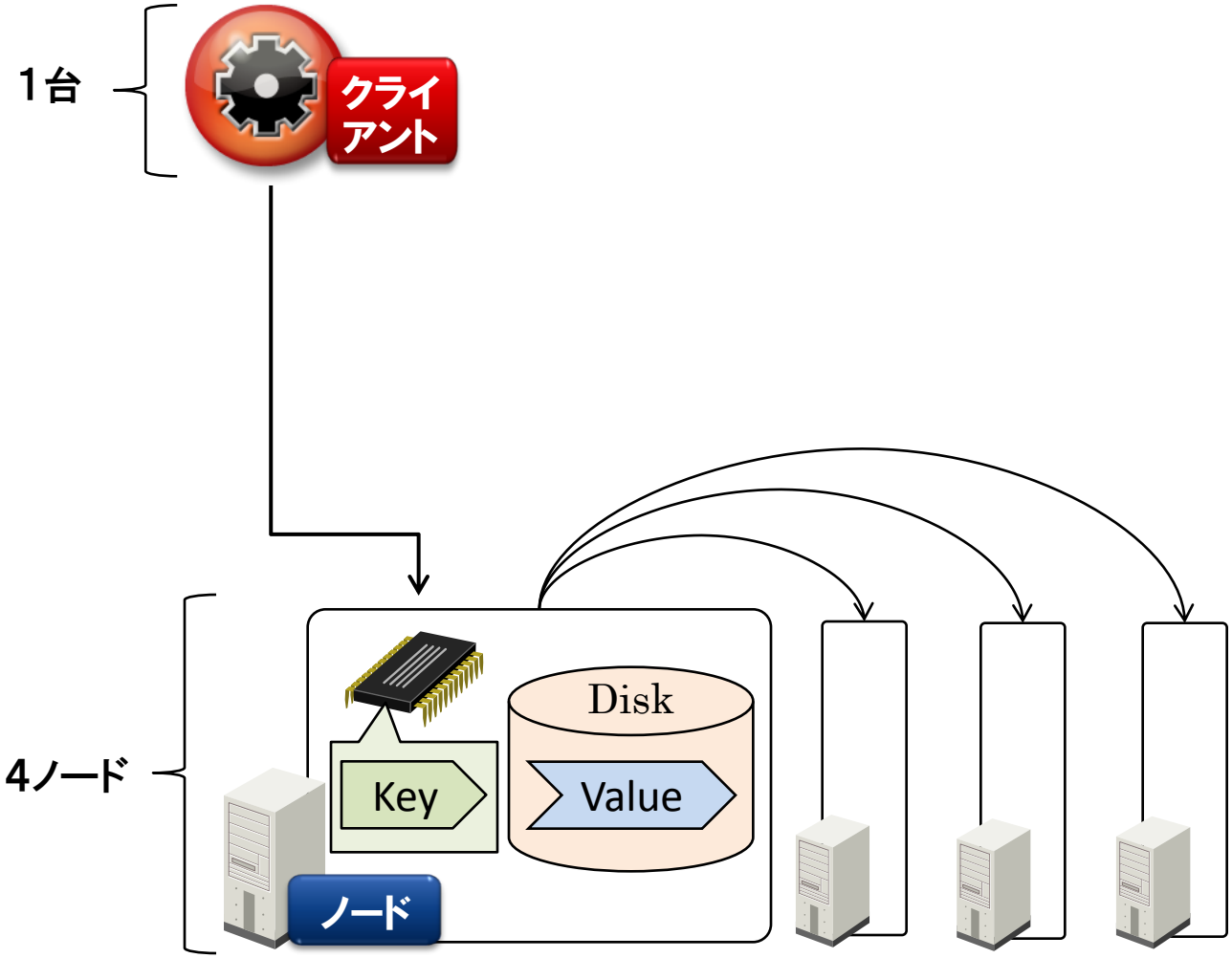
観点1：同時書込み性能：同時アクセス数を増加



3.5. 検証④ Riak –【メモリ・ディスク併用】

HITACHI
Inspire the Next

構成見直し前	P2P



3.5. 検証④ Riak –【メモリ・ディスク併用】

HITACHI
Inspire the Next

構成見直し後

P2P

観点1：同時書込み性能

同時アクセス数を増加

観点2：メモリ消費時の性能

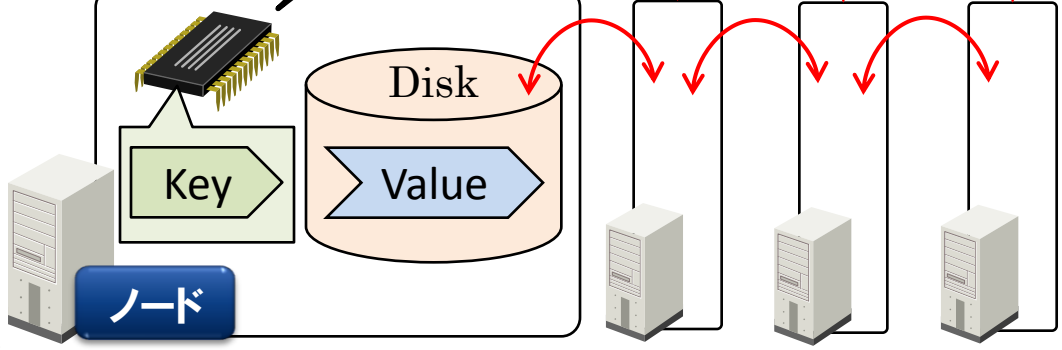
メモリが無くなるまで長時間書込み

2台

クライアント

均等に
負荷分散

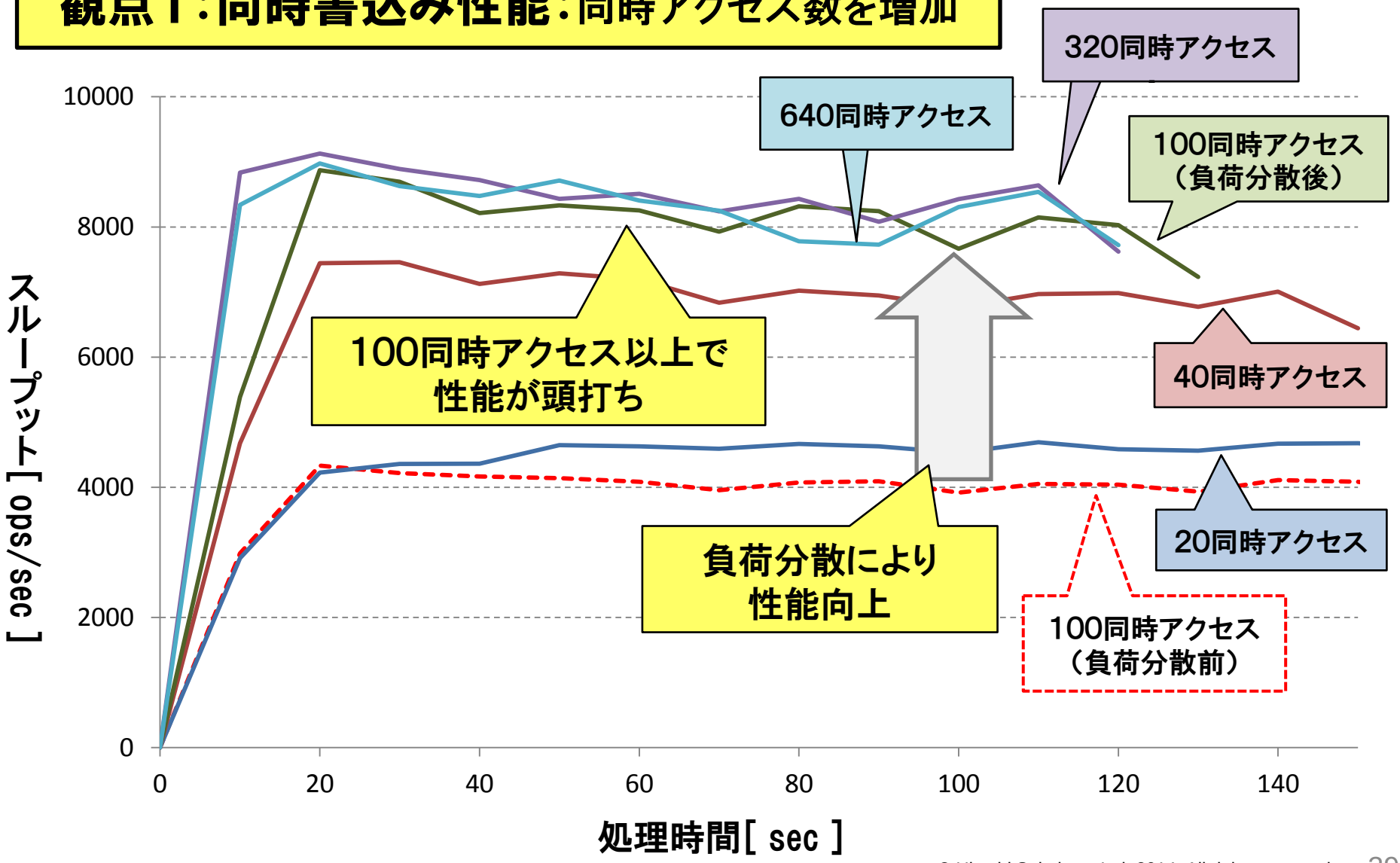
4ノード



3.5. 検証④ Riak –【メモリ・ディスク併用】

HITACHI
Inspire the Next

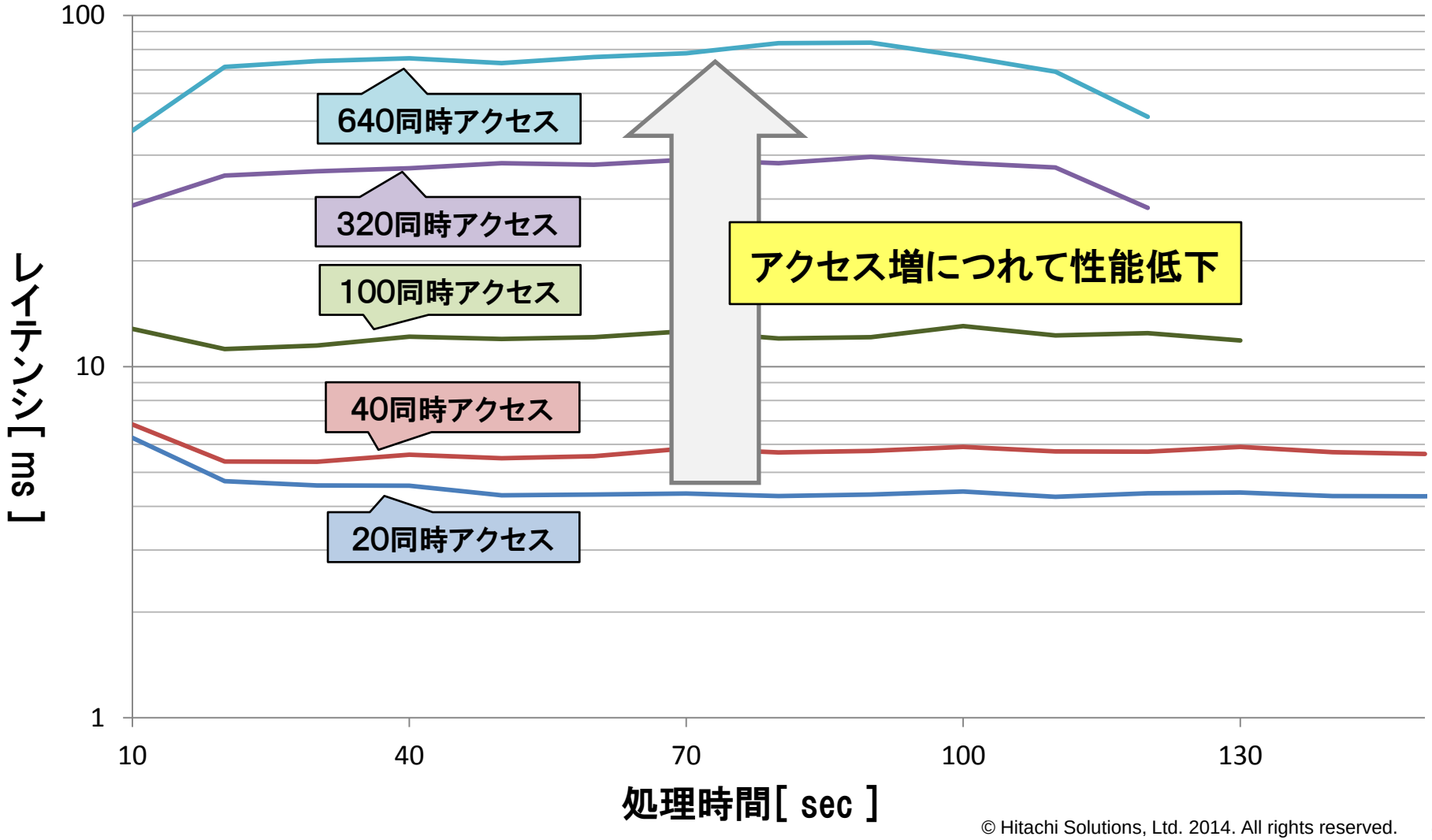
観点1：同時書込み性能：同時アクセス数を増加



3.5. 検証④ Riak –【メモリ・ディスク併用】

HITACHI
Inspire the Next

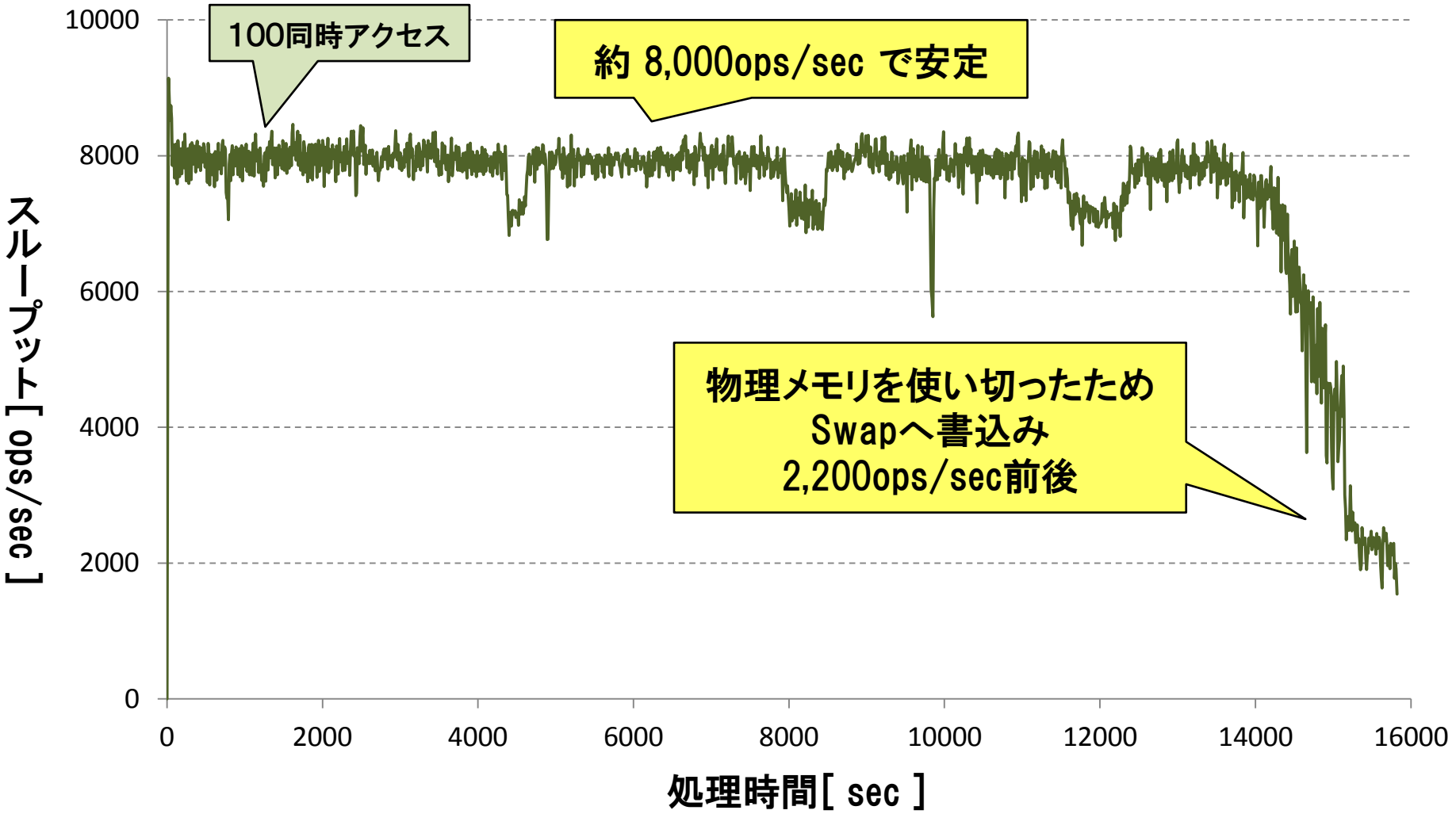
観点1：同時書込み性能：同時アクセス数を増加



3.5. 検証④ Riak –【メモリ・ディスク併用】

HITACHI
Inspire the Next

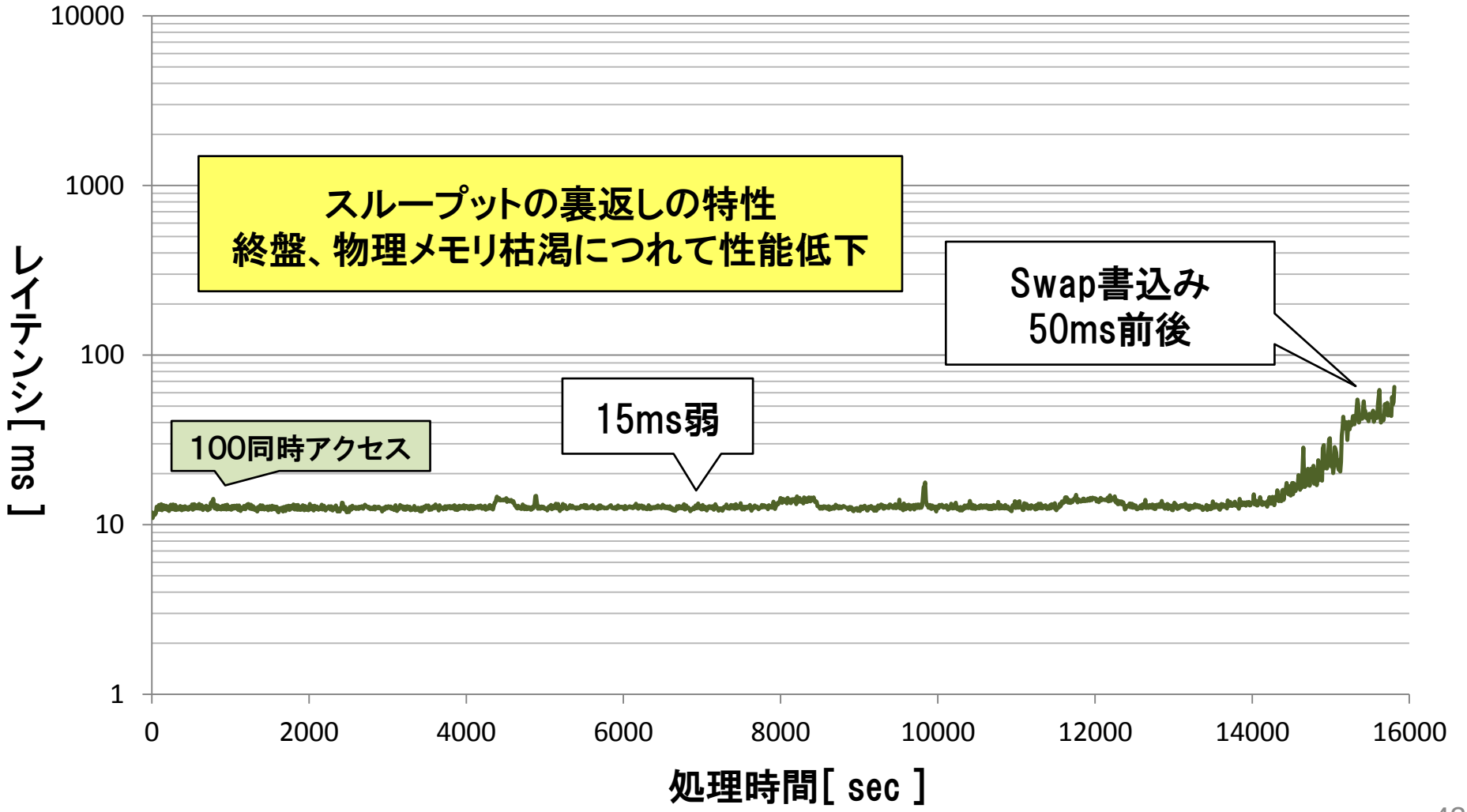
観点2：メモリ消費時の性能：メモリが無くなるまで長時間書込み



3.5. 検証④ Riak –【メモリ・ディスク併用】

HITACHI
Inspire the Next

観点2：メモリ消費時の性能：メモリが無くなるまで長時間書込み



3.5. 検証④ Riak -【メモリ・ディスク併用】

HITACHI
Inspire the Next

わかったこと

１）同時アクセス性能

同時アクセス数が増加すると、データ振分け処理が
ボトルネック。クライアントアクセスの負荷分散が必要。

２）メモリ消費時の性能

物理メモリを使い切るまでは安定した性能。
物理メモリを使い切るとSwap書込みとなり、
大きく性能低下。

３）スループット（中盤の安定状態）

平均約 8,000 ops/sec（4ノードクラスタ）

HITACHI
Inspire the Next

4. 考察

4.1. センサーデータ蓄積で使う時のポイント

HITACHI
Inspire the Next

【ディスク保存】

ディスク性能・クラスタ台数を要検討

ディスクI/Oがボトルネック。スケールアウトで性能向上するので、
性能要件に応じて検討すること。

【メモリ・ディスク併用】

1）クライアントからKVSへの書込み負荷分散が重要

2）残メモリ量に要注意

1) KVSのデータ振分け処理がボトルネックになりやすいので、
負荷分散すること。

2) メモリがなくなると性能が大幅ダウン。
メモリの監視、スケールアウトのタイミングなど、運用を検討すること。

日立ソリューションズ

【補足】永続化について

HITACHI
Inspire the Next

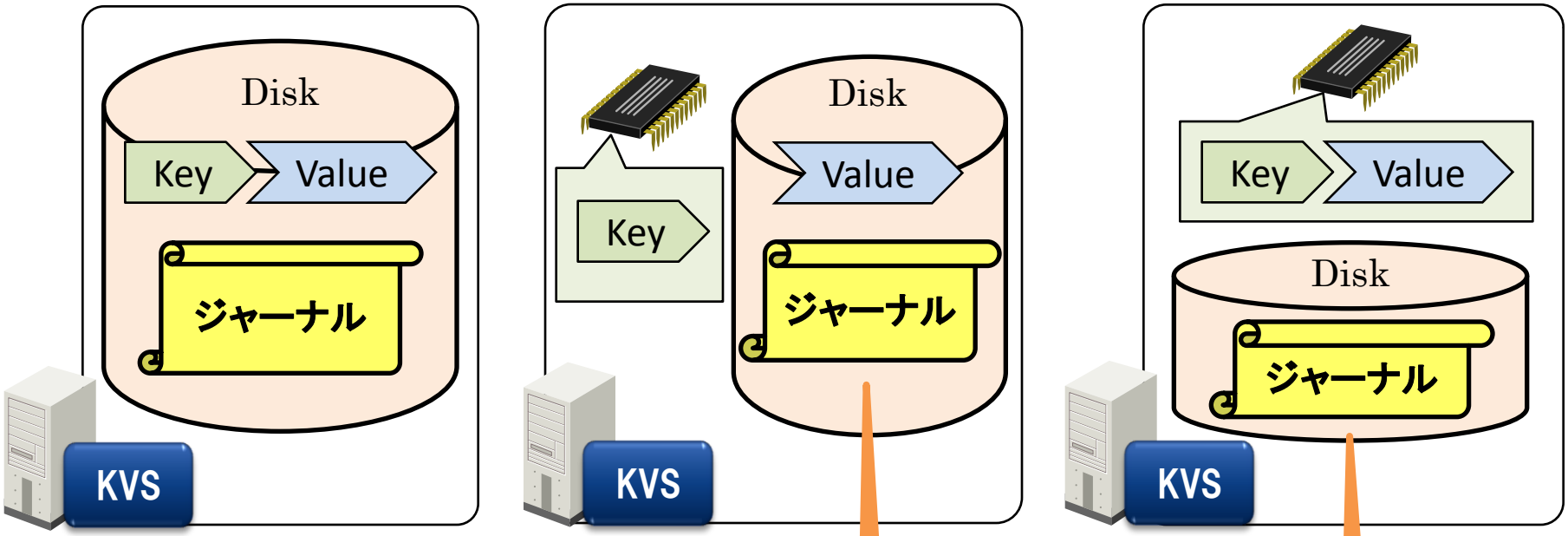
※データの保存先にかかわらず、永続化は可能

データ操作の履歴ログ(ジャーナル)を作成し永続化

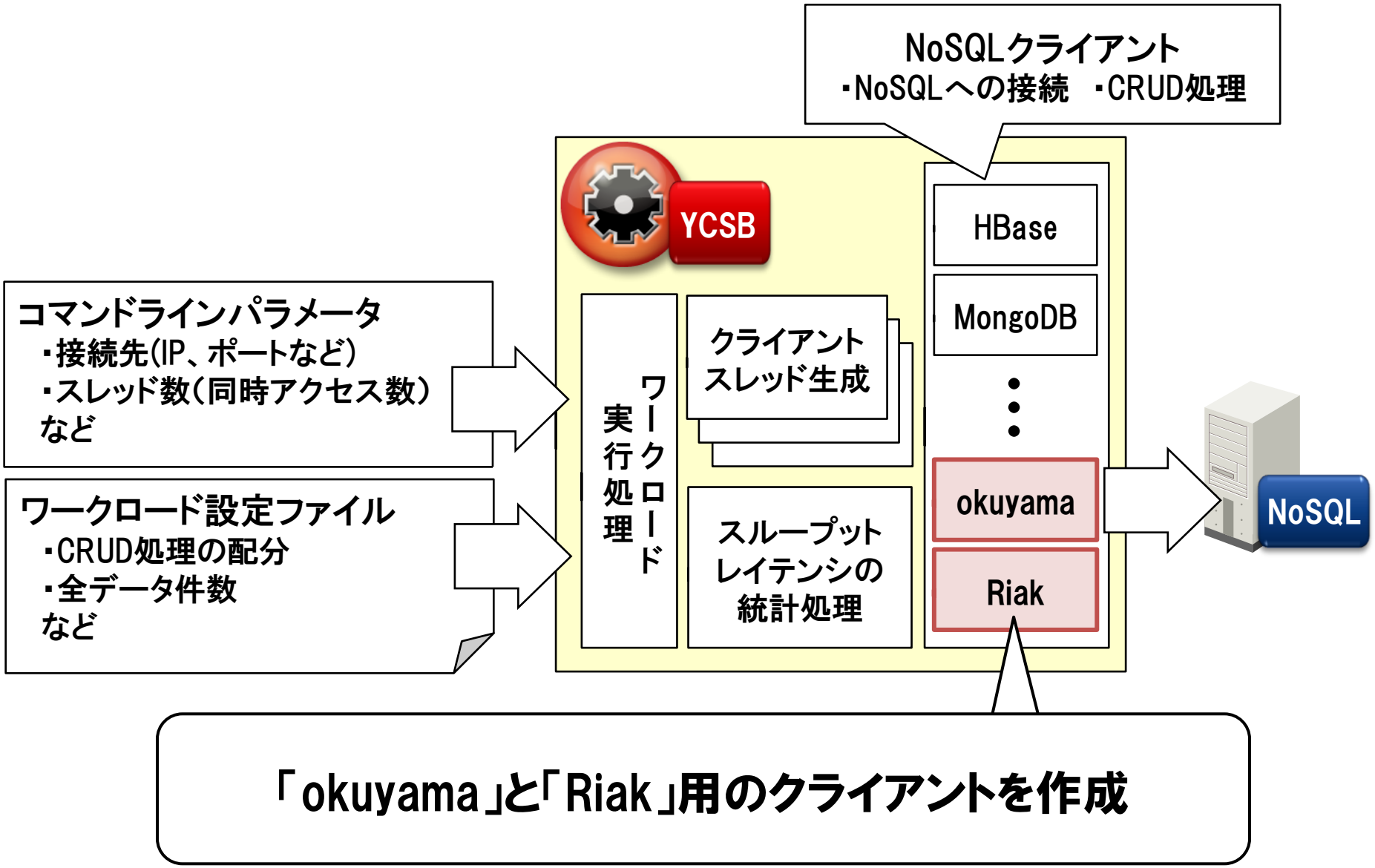
・パターン1
KeyとValueをディスク保存

・パターン2
Keyをメモリ、
Valueをディスク保存

・パターン3
KeyとValueをメモリ保存



【補足】YCSBの構成



参考：http://labs.yahoo.com/files/ycsb-v4.pdf

【補足】okuyama, Riak クライアント

HITACHI
Inspire the Next

KVSのJavaライブラリを利用し、クライアントを作成

他NoSQL用のクライアントを流用

```
public int insert(String table, String key, HashMap<String, ByteIterator> values) {
    com.mongodb.DB db = null;
    try {
      db = mongo.getDB(database);
      db.requestStart();
      DBCollection collection = db.getCollection(table);
      DBObject r = new BasicDBObject().append("_id", key);
    for(String k: values.keySet()) {
      r.put(k, values.get(k).toArray());
    }
      WriteResult res = collection.insert(r,writeConcern);
      return res.getError() == null ? 0 : 1;
    } catch (Exception e) {
      System.err.println(e.toString());
      return 1;
    } finally {
      if (db!=null)
      {
        db.requestDone();
      }
    }
  }
```

okuyamaとRiak用にメソッドを書き換え

```
public int insert(String table, String key, HashMap<String, ByteIterator> values) {
    try {
      boolean setResult = okuyamaClient.setValue(key, key);
      if (setResult) {
        System.out.println("setKey = "+key);
        } else {
          System.out.println("setValue Result = [Server Error]");
          System.exit(1);
        }
    } catch(OkuyamaClientException oc) {
      oc.printStackTrace();
    }
    return 0;
  }
```

- okuyama、Riakクライアント共に80行程度
- 書込むデータは、KeyとValue共に約25バイト

書込みベンチマーク用のため、KVS接続処理とInsert処理のみ作成

- okuyama ： okuyama本体に同梱のライブラリ
- Riak ： 公式サポートのJavaライブラリ　※入手先:【https://github.com/basho/riak-java-client/tree/1.4.2】

【補足】その他 注意点

HITACHI
Inspire the Next

- **ベンチマークの間、リソース監視データを出力**

  リソース状況の確認のため、
  YCSBおよびKVSサーバ上でsarコマンドを実行し、
  ベンチマーク中のリソースのデータを出力した。

- **KVSはテスト毎に再インストール**

  KVS内のテストデータ削除のため、
  1テストケース実施ごとに、
  KVSのアンインストールと再インストールを実施した。