HITACHI
Inspire the Next

Hadoop & Spark性能検証

～HiveとSpark SQLによる集計処理の比較～

2016/9/1

株式会社 日立製作所 OSSソリューションセンタ

木下 翔伍

HITACHI
Inspire the Next

Contents

自己紹介

HITACHI
Inspire the Next

木下 翔伍 / Kinoshita Shogo

1990-2011年
- ずっと京都在住

2012年
- (株)日立製作所　入社

2015年
- Pentahoソフトウェアサポート業務と案件対応に従事
- この頃からビッグデータに関わる

2016年
- OSSソリューションセンタ
  ビッグデータ関連OSS(Hadoop、Spark)を活用したソリューション開発業務と**案件対応**に従事

電力事業者への適用を想定したユースケースのもとで実施したHadoopとSparkの性能検証結果を報告

- Hadoop Summit 2016 San Jose に参加

HITACHI
Inspire the Next

1. ビッグデータ利活用の取り組み
[電力事業者の想定適用例]

1-1 電力事業者をとりまく環境およびニーズの例

HITACHI
Inspire the Next

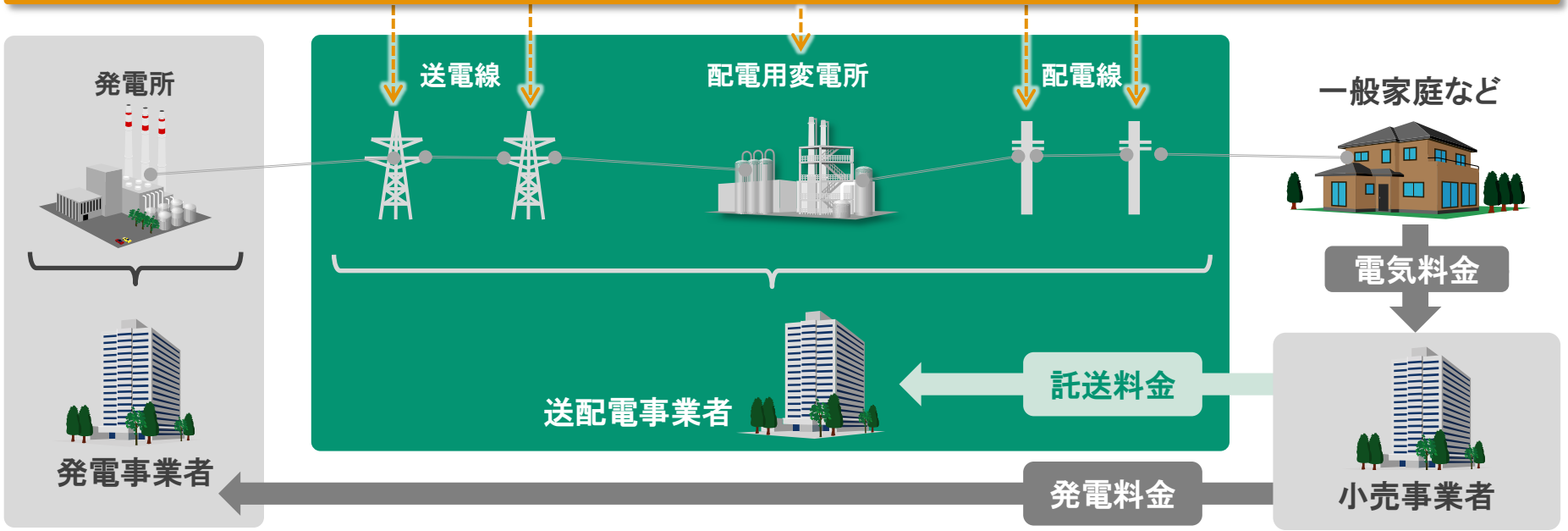
◆ 電力小売自由化

- 既存電力事業者は、競争的な自由市場への移行と対応が必要
- 国から託送料金の見直しを検討するよう要望あり

◆ 送配電設備に関するコストカットの必要性が高まる

- 従来、送配電設備は一定期間使用すれば、状態にかかわらず取り替え
- 時間基準から状態基準への設備管理へ移行し、状態の良い設備は使い続けたい

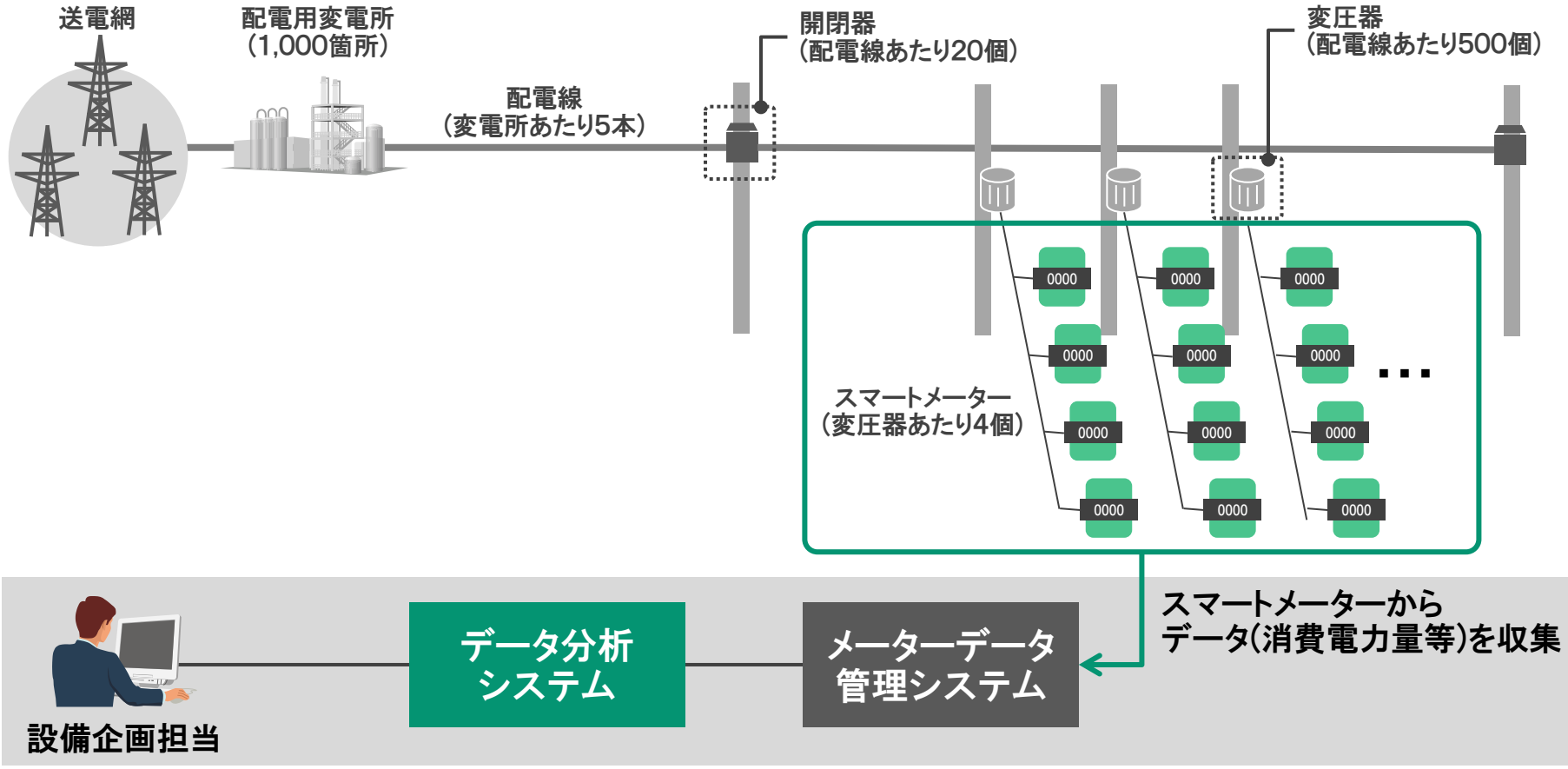
設備個体ごとの運用状況を細かに把握したいというニーズがある



1-2 送配電事業向けビッグデータ利活用の取り組み

HITACHI
Inspire the Next

◆ 送配電事業のニーズに対応した取り組み



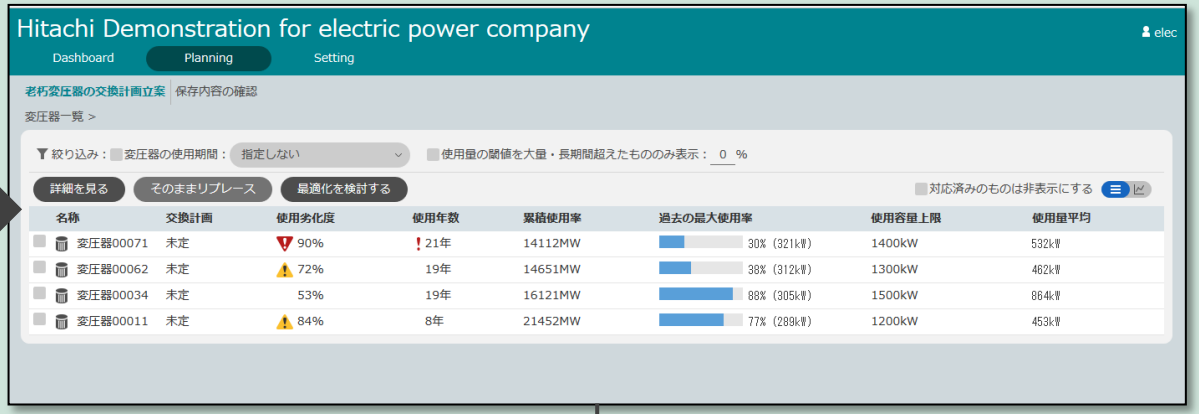
**スマートメーターから収集したデータを
ビッグデータとして分析して設備状況を把握**

1-3 システム構成

HITACHI
Inspire the Next

データ分析システム

設備企画担当



データ分析アプリケーション

本セッションのメイン

データ処理基盤（Hadoop、Spark）

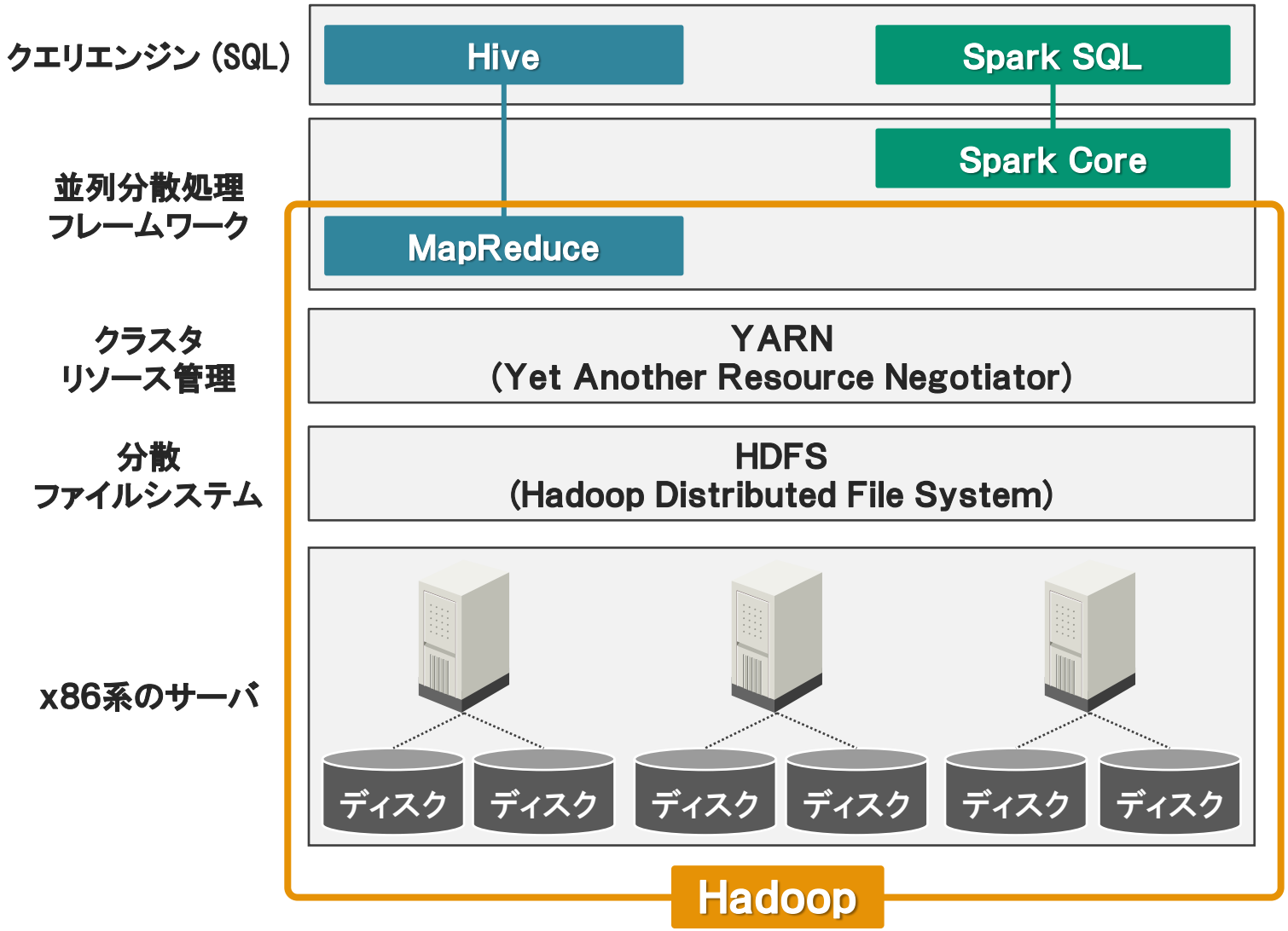
メーターデータ管理システム

HITACHI
Inspire the Next

2. Hadoop/Spark の基礎

2-1 分散処理基盤 Hadoop / Spark

HITACHI
Inspire the Next



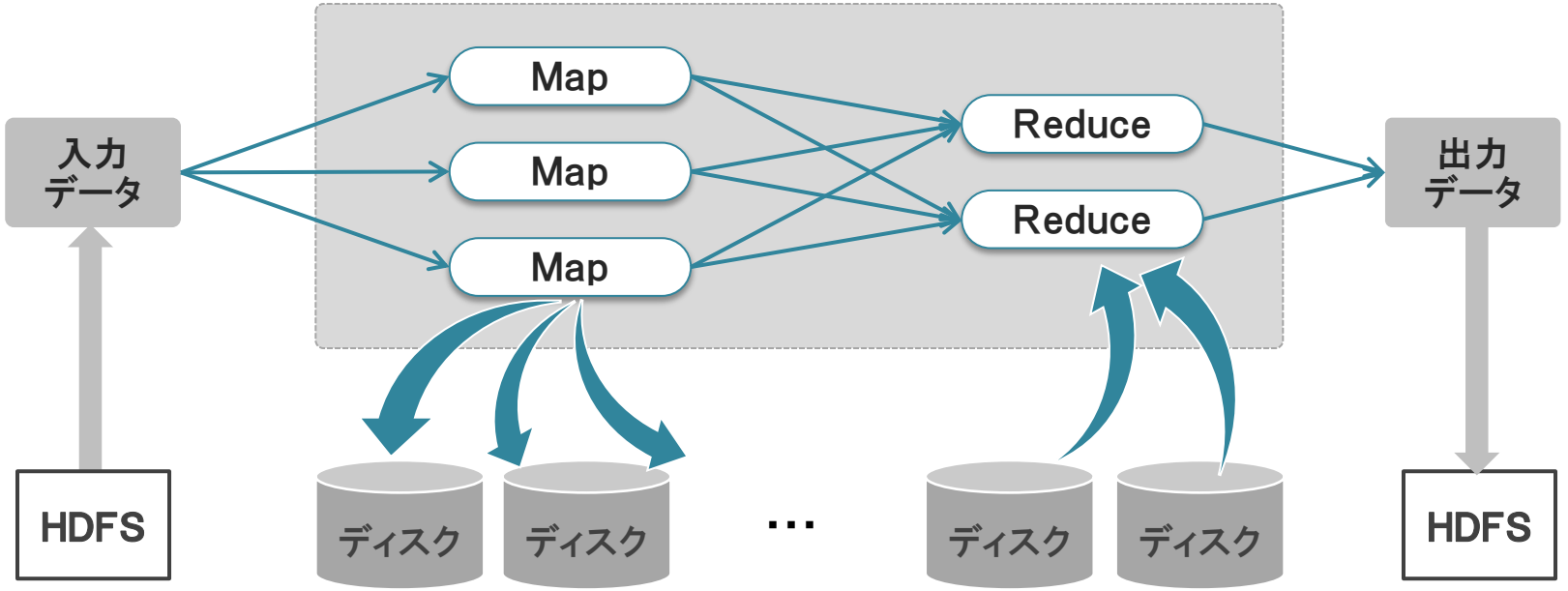
ビッグデータを処理する用途で注目を集める

2-2 並列分散処理フレームワーク MapReduce

HITACHI
Inspire the Next

◆ MapReduceの処理方式

特徴	• 読み出した入力データを複数に分割して並列に処理 • 処理中に発生するディスクアクセスを並列化して実行



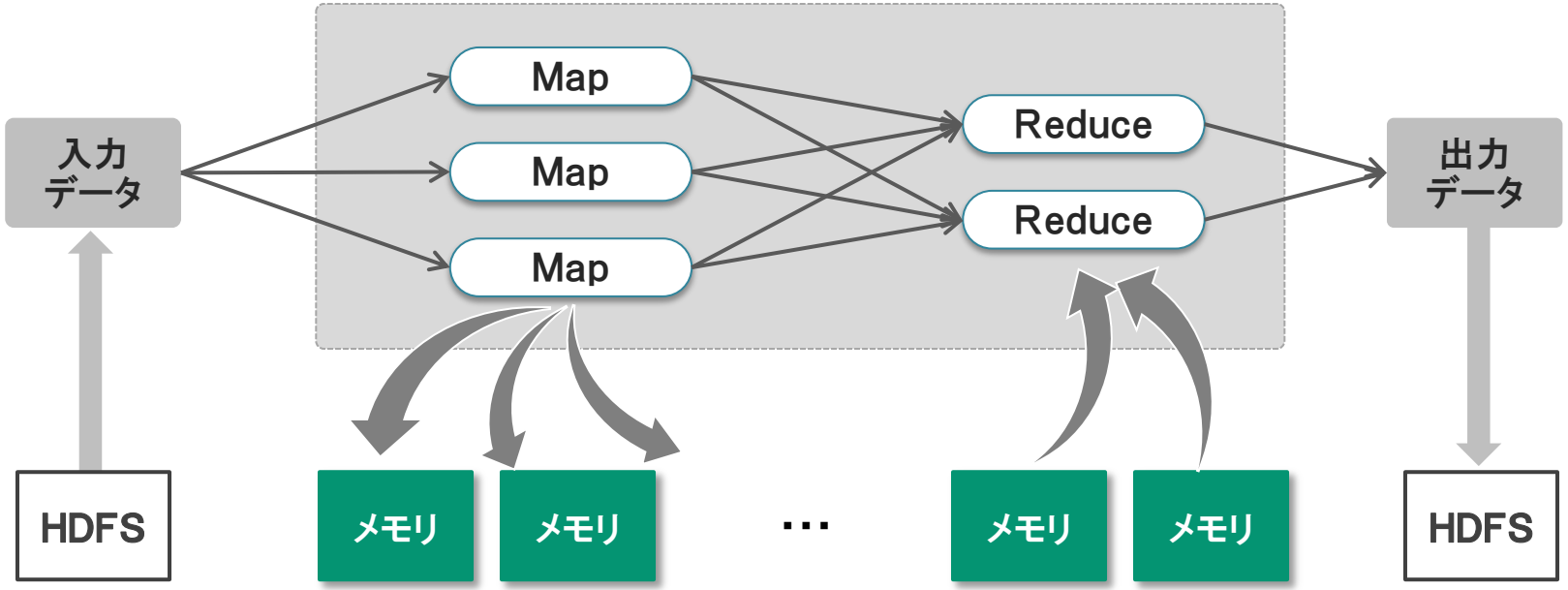
処理とディスクアクセスを並列化することで性能向上

2-3 インメモリ分散処理フレームワーク Spark

HITACHI
Inspire the Next

◆ Sparkの処理方式

特徴	• MapReduceと同様に入力データを複数に分割して並列に処理 • 処理中に扱うデータはメモリ上で保持



メモリ上でデータを扱うことでMapReduceよりも高速動作

HITACHI
Inspire the Next

3. HiveとSpark SQLを使った性能検証

3-1 ユースケース

HITACHI
Inspire the Next

◆ 送配電事業者のニーズと解決手段（再確認）

ニーズ	設備投資効率化のために設備の運用状況を細かく把握したい
解決手段	スマートメーターから収集したデータを分析して設備状況を把握する

◆ 想定される分析ケース

管理上のニーズ	分析でやるべきこと
取替が必要な設備を見つけ出したい	負荷が集中または増加傾向にある設備を探す
取替が必要な大まかな時期を割り出したい	短期的な傾向と長期的な傾向を比較する
新しく取替える設備の機種を選定したい	消費電力量のピーク時間を抽出する

◆ 想定される要件

- メーターからの30分ごとにデータ収集（メーター1台から1日48個のデータを収集）
- 収集した新しいデータを分析結果に反映できるとより良い

3-2 検証内容

HITACHI
Inspire the Next

◆ 検証観点

想定要件： 新しいデータを分析できるとよい

- 分析のたびに収集した電力消費量データをすべて集計すると処理時間が掛かる
- あらかじめ内部で、適当な単位で電力消費量を集計したデータを持つことで、分析処理の軽減が必要
- その集計データの作成目標時間はメーターからのデータ収集間隔の30分以内であるべき

MapReduceとSparkは、集計処理の目標時間30分以内を満たせるかを性能検証する

◆ 測定項目

分析でやるべきこと	集計観点
負荷が集中または増加傾向にある設備を発見する	配電設備
短期的な傾向と長期的な傾向を比較分析する	期間
消費電力量のピーク時間を抽出する	時間帯

電力消費量データの集計単位

・配電系全体　・変電所ごと　・開閉器ごと
・変圧器ごと　・メーターごと

配電系統の負荷を集計する期間

・1日　・1か月(30日)　・1年(365日)

集計対象の時間帯(データ)

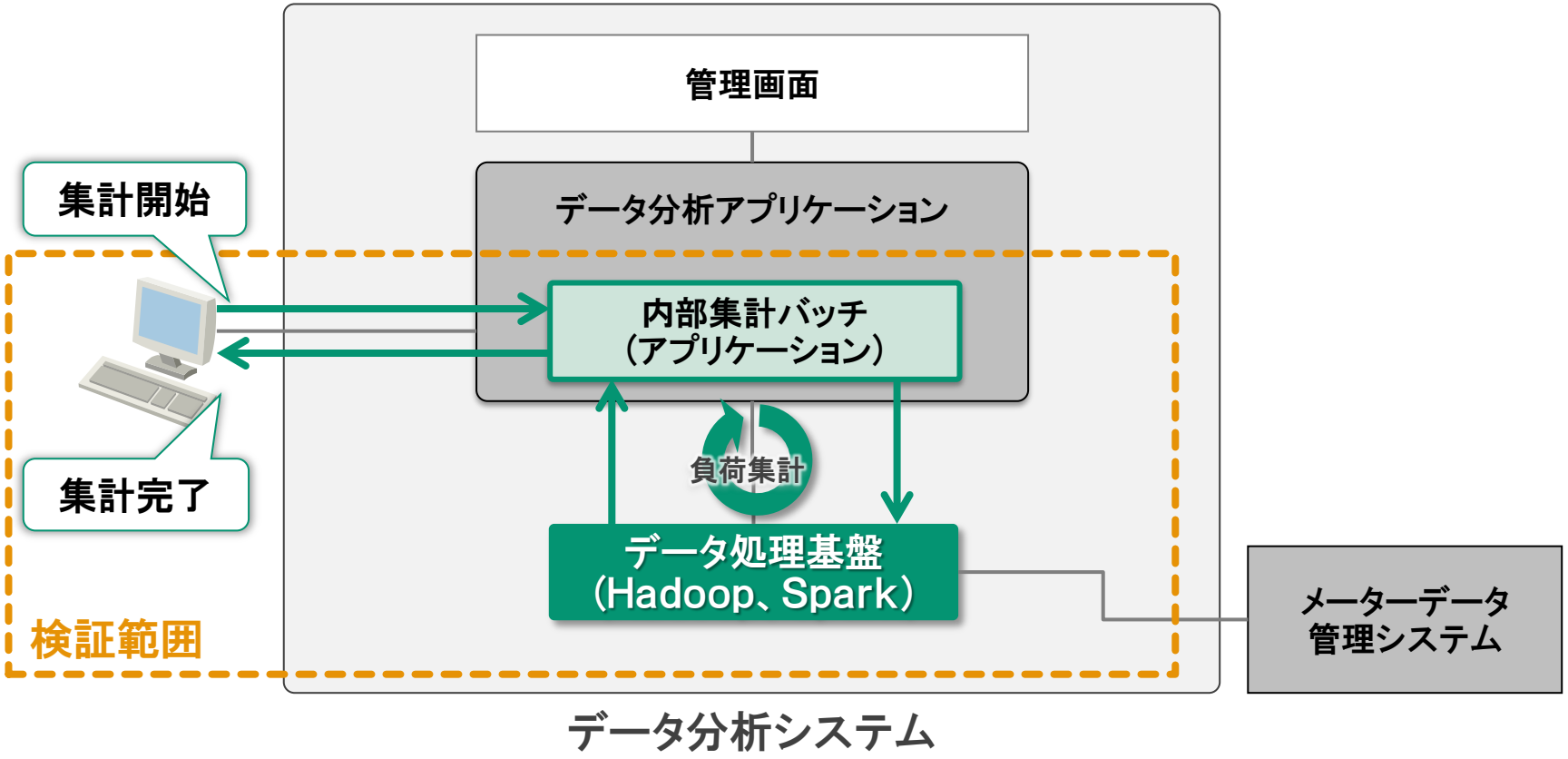
・特定の30分間　・24時間すべて

3-3 検証範囲

HITACHI
Inspire the Next

◆ 検証範囲

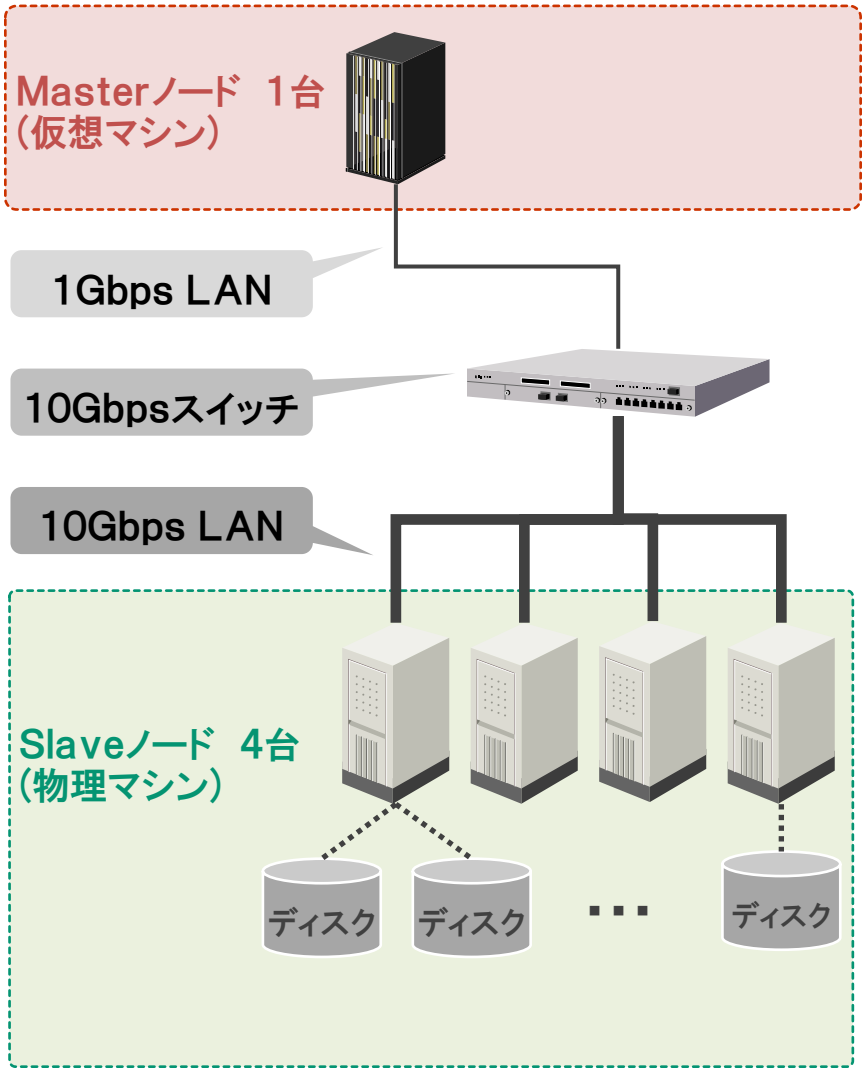
測定内容	電力消費データを内部で集計するバッチを開始してから完了するまでの所要時間



3-4 検証環境

HITACHI
Inspire the Next

◆ マシン構成



◆ スペック

項目	Masterノード
CPUコア数	2 コア
メモリ容量	8 GB
ディスク 1個の容量	80 GB
ディスク数	1 個
ディスク 総容量	80 GB

項目	Slaveノード単体	Slave全体
CPUコア数	16 コア	64 コア
メモリ容量	128 GB	512 GB
ディスク 1個の容量	900 GB	-
ディスク数	6 個	24 個
ディスク 総容量	5.4 TB (5,400 GB)	21.6 TB (21,600 GB)

3-5 検証条件

処理性能に影響があるパラメータを変動させて集計処理に要する時間を測定

◆ データソース

HDFSに格納するファイル形式

- CSV （テキストファイル）
- ORCFile （列指向ファイル）

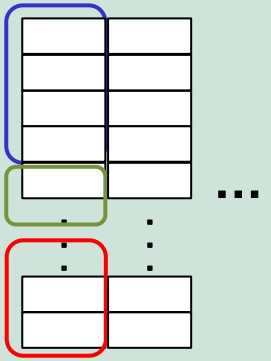
◆ メモリサイジング

クラスタのメモリ割当容量

- 1 Slaveノードあたり最大128GB
- Slaveノードは全4台
  - 512GB (128GB/台)
  - 256GB (64GB/台)
  - 128GB (32GB/台)
  - 96GB (24GB/台)
  - 80GB (20GB/台)
  - 64GB (16GB/台)

ORCFile とは

- 複数レコード(行)をまとめて固定長のカラム(列)単位で管理
  → 列単位で読み込みが可能
- ファイル作成時に圧縮機能使用可能
  → zlib もしくは snappy

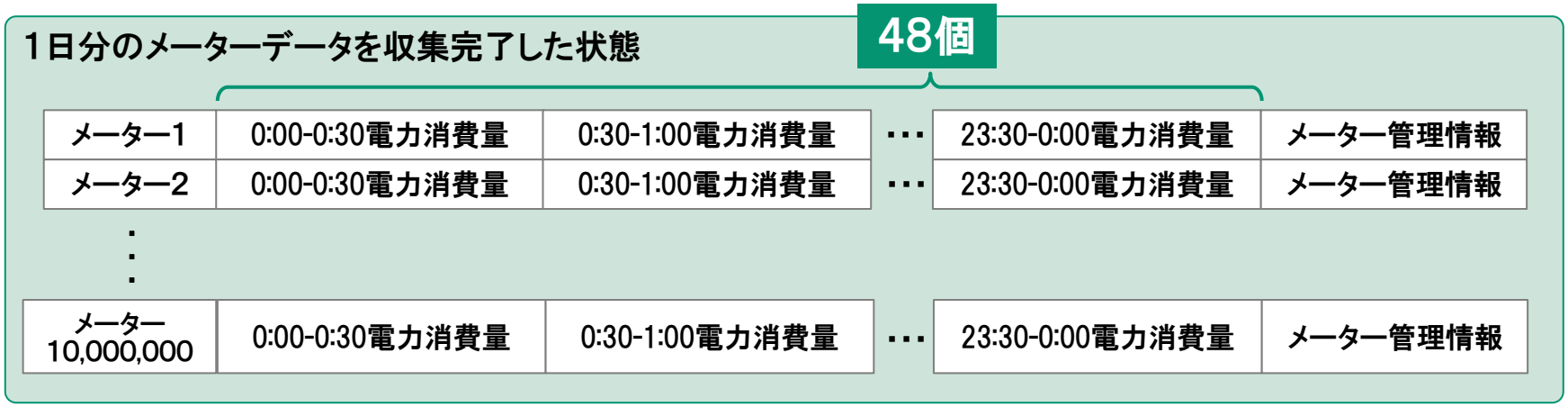


3-6 データセット

HITACHI
Inspire the Next

◆ メーターデータ

- スマートメーターから30分おきに収集した1日分(48個)の電力消費量データ
- メーター管理情報



データサイズは次のとおり

#	集計期間	レコード数	サイズ (CSV)	サイズ (ORCFile)
1	365日 (1年)	36億5,000万	2.475 TB	1.325 TB
2	30日 (1ヶ月)	3億0,000万	0.205 TB	0.158 TB
3	1日	1,000万	0.007 TB	0.005 TB

検証内容（要約・再確認）

HITACHI
Inspire the Next

◆ 検証観点

MapReduceとSparkは、集計処理の目標時間30分以内を満たせるかを性能検証する

◆ 検証範囲

測定内容	電力消費データを内部で集計するバッチを開始してから完了するまでの所要時間

◆ 測定項目

電力消費量データの集計単位

・配電系全体　・変電所ごと　・開閉器ごと
・変圧器ごと　・メーターごと

配電系統の負荷を集計する期間

・1日　・1か月(30日)　・1年(365日)

集計対象の時間帯(データ)

・特定の30分間　・24時間すべて

◆ 検証条件

HDFSに格納するファイル形式

- CSV （テキストファイル）
- ORCFile （列指向ファイル）

クラスタのメモリ割当容量

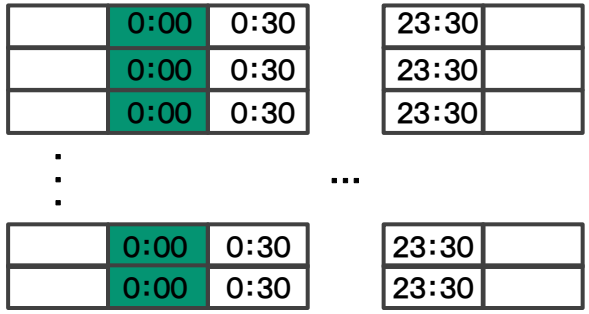
- 512GB (128GB/台)
- 256GB (64GB/台)
- 128GB (32GB/台)
- 96GB (24GB/台)
- 80GB (20GB/台)
- 64GB (16GB/台)

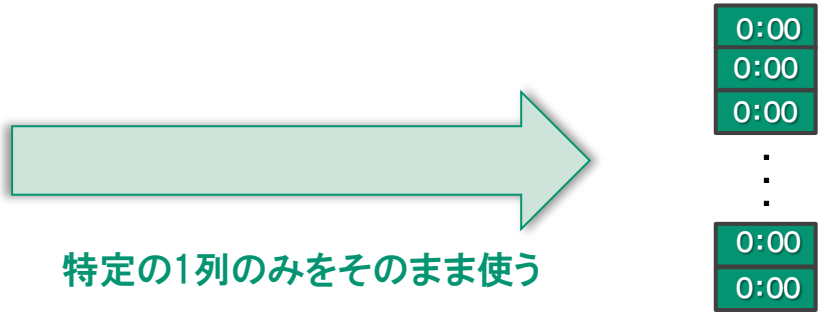
3-7 集計対象の時間帯による処理方式の違い

HITACHI
Inspire the Next

◆ 0.5時間(1回分)の場合

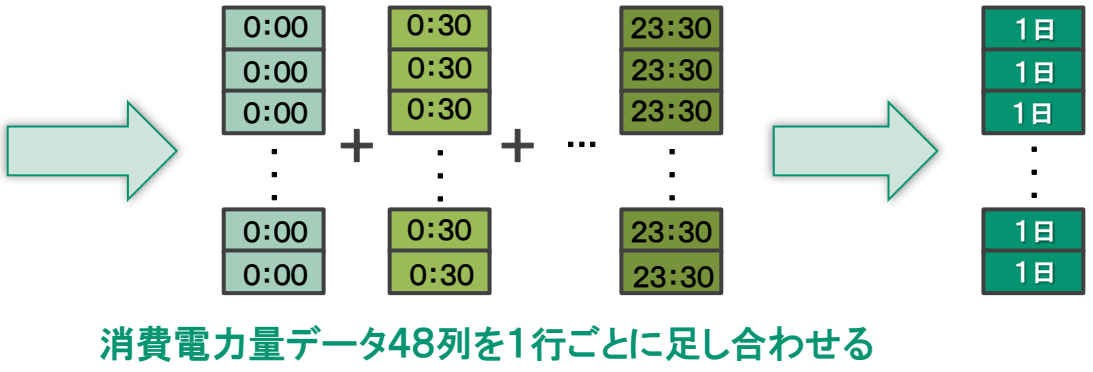
特定列（特定時間帯の消費電力量データ）のみを使用する





◆ 24時間(48回分)の場合

すべての消費電力量データを足し合わせてから使用する

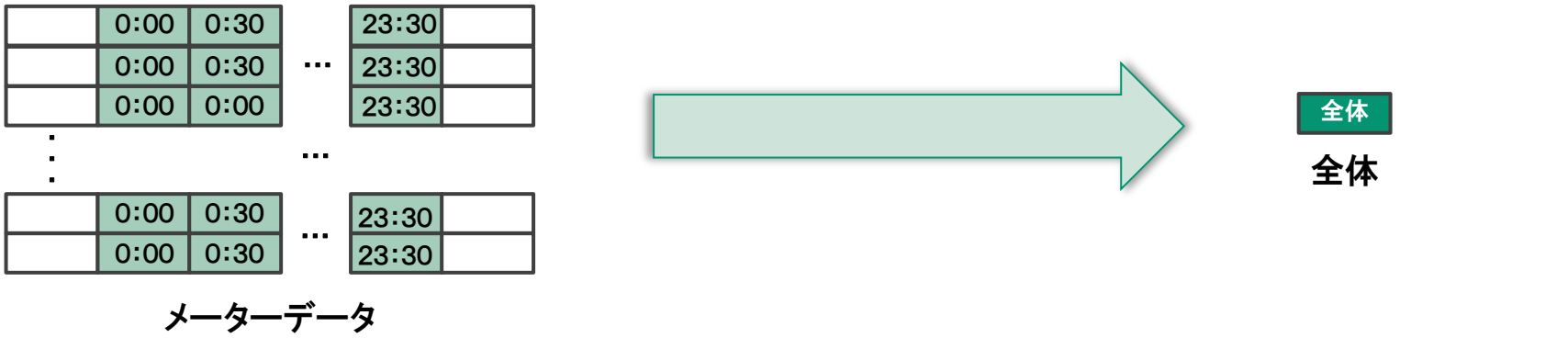


集計対象の時間帯を変えることで集計に用いる列が異なる

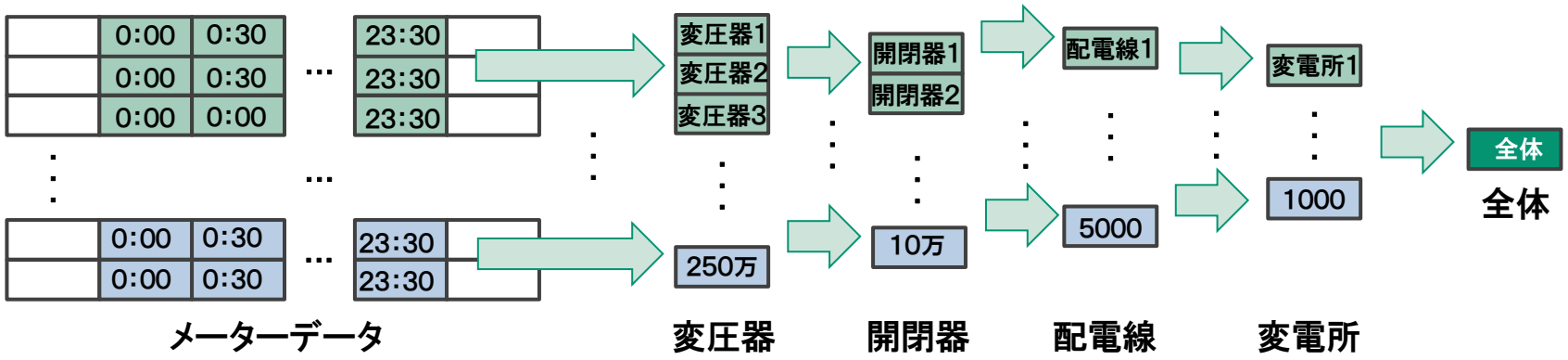
3-8 集計単位による処理方式の違い

HITACHI
Inspire the Next

◆ 配電系統全体の場合　1回の処理ですべての行の消費電力量データを集計する



◆ 設備ごとの場合　設備ごとに1個ずつ順番に消費電力量データを集計する



集計単位を変えることで処理回数が異なる

3-9 実施事項

HITACHI
Inspire the Next

◆ 測定時に設定するパラメータ

集計単位	集計期間	ファイル形式	集計対象の時間帯	メモリ割当容量
• 配電系統全体 • 設備ごと	• 1日 • 30日 • 365日	• CSV • ORCFile	• 0.5 時間 • 24 時間	• 512GB • 256GB • 128GB • 96GB • 80GB • 64GB

◆ ファイル形式の違いによる性能検証

測定条件	変動パラメータ
・集計単位　配電系統全体 ・メモリ割当容量　512 GB	集計期間 集計対象の時間帯

◆ 集計単位の違いによるHive/Sparkの性能検証

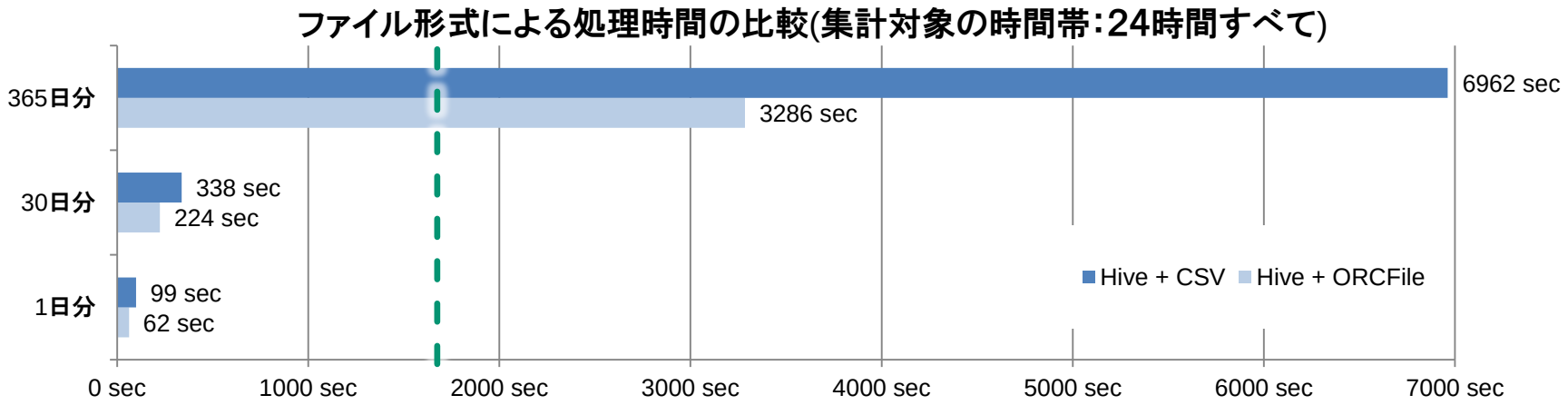
測定条件	変動パラメータ
・ファイル形式　ORCFile ・メモリ割当容量　512 GB	集計期間 集計対象の時間帯

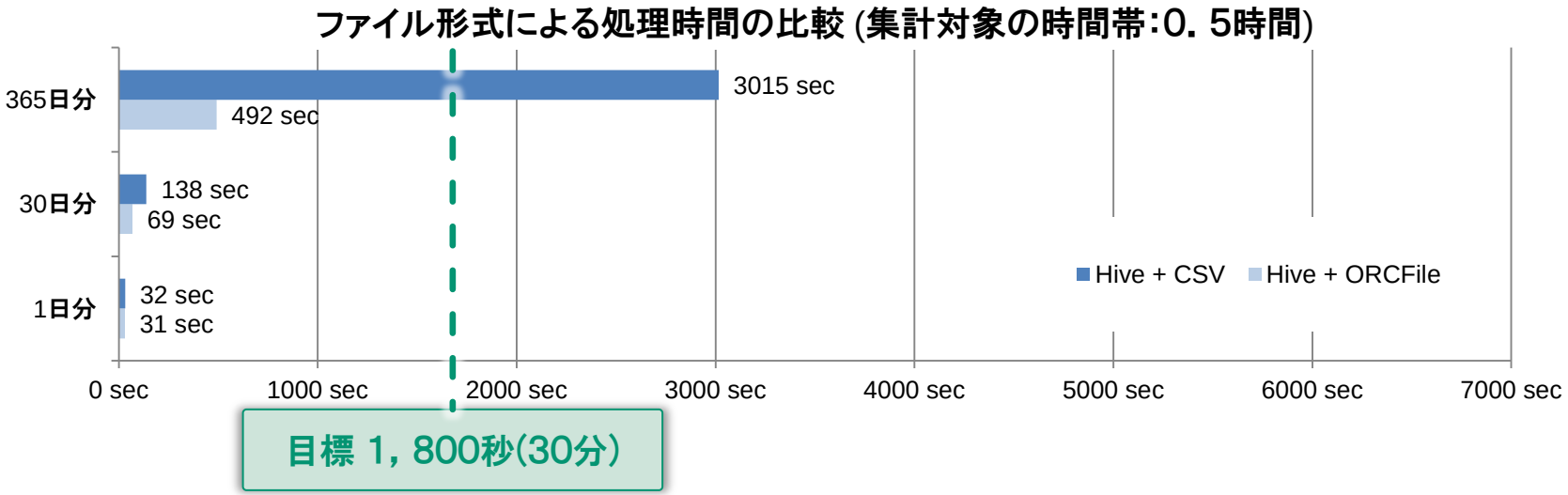
◆ メモリ割当容量の違いによるHive/Sparkの性能検証

測定条件	変動パラメータ
・集計期間　365日 ・集計対象の時間帯　24時間 ・ファイル形式　ORCFile	集計単位

3-10 ファイル形式 - Hive

HITACHI
Inspire the Next



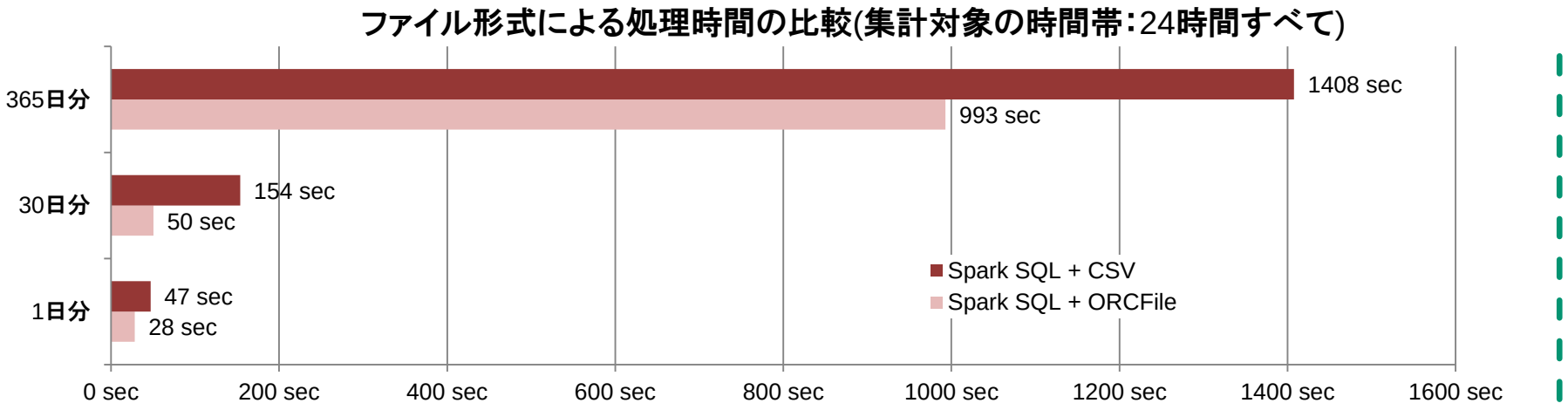


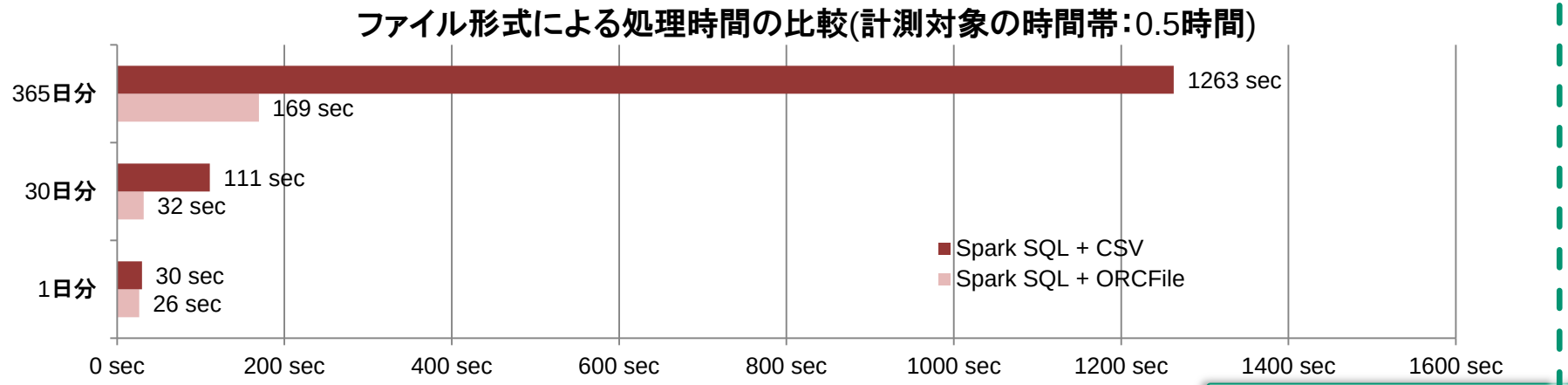
目標 1, 800秒(30分)

- 目標時間以内に処理完了できないことがある
- CSVよりもORCFileを利用した場合の方が処理性能が高い

3-11 ファイル形式 - Spark SQL

HITACHI
Inspire the Next





目標 1，800秒(30分)

- どちらの場合でも目標時間以内に処理完了
- CSVよりもORCFileを利用した場合の方が処理性能が高い

3-12 ファイル形式の違いにおける考察

HITACHI
Inspire the Next

◆ 結果

HiveとSpark SQLはともに
CSVよりもORCFileをデータ入力に用いるほうが処理性能を向上させることができる

◆ CSVよりもORCFileの方が処理時間が短いのは何故か

ORCFileを利用したときの測定結果

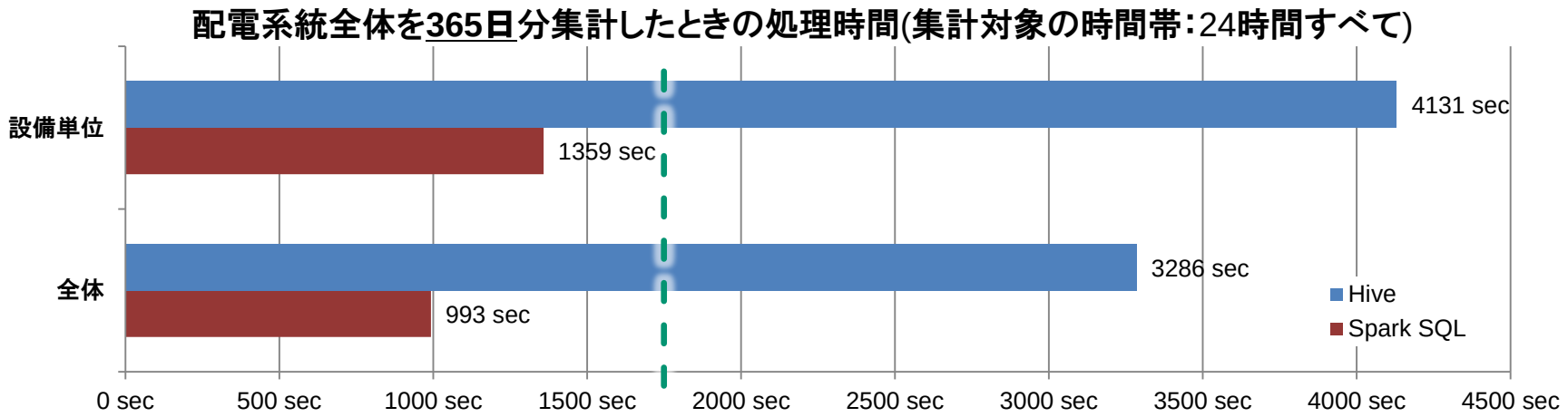
- すべての測定で、CSVよりも処理時間が短い
- 処理対象のデータが多いほど短縮できる時間が大きい
- 集計対象の時間帯が24時間よりも0.5時間のメーターデータを集計する処理の方が時短効果が高い

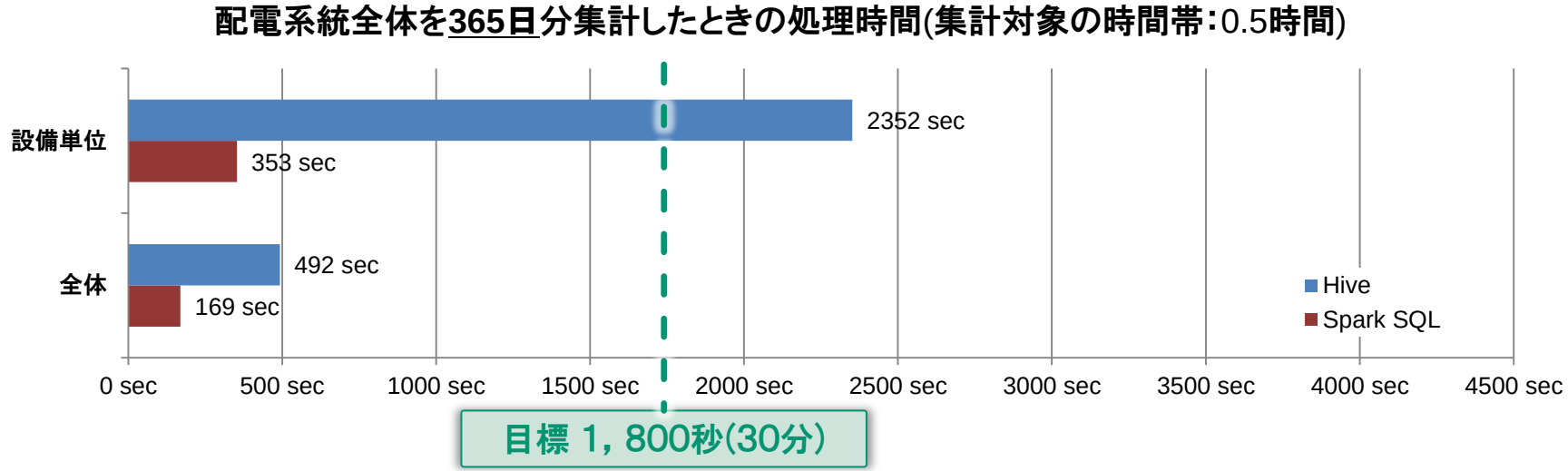
対応するORCFileの特徴

- ファイル圧縮
- さらに
- 特定列のみの読み出し

**ORCFileの特徴によって
データの読み出し量（ディスクアクセス時間）が低減されたことで
処理時間を短縮できた**

3-13 集計単位





- Hiveでは目標時間以内に処理完了できないことがある
- HiveよりもSpark SQLの方が処理時間が短い

3-14 集計単位の違いにおける考察

HITACHI
Inspire the Next

◆ 結果

HiveとSpark SQLでは
Spark SQLを用いるほうが高い処理能力を得られる

◆ 配電系統よりも設備ごとで集計したほうが短縮できる時間が大きいのは何故か

集計単位による処理方式の違い（再確認）

- 配電系統全体の場合・・・　1回の処理ですべての行の消費電力量データを集計する
- 設備ごとの場合　・・・　設備ごとに1個ずつ順番に消費電力量データを集計する

集計単位を変えることで処理回数が異なる

ディスクアクセスがメモリアクセスに置き換わった処理の回数が多いため

配電系統全体に比べて設備ごとの集計で短縮できた時間は異なる

- 集計期間365日のとき、最大6.1倍の差

HITACHI
Inspire the Next

4. まとめ

4-1 ノウハウ

HITACHI
Inspire the Next

◆ ORCFile形式に適したデータ

ORCFile形式でのファイル作成には時間が掛かる
→ 1年分のメーターデータ作成で約3時間

蓄積されて参照される用途のデータが適する
（値が何度も更新されるデータは不適）

◆ Hiveでのカラム集計方法

集計対象の時間帯が24時間のとき、
1レコードごとに電力消費量データ48個を足し合わせる処理を実装する方法は2種類ある

[1]Hive Streaming機能で別プロセスから
スクリプトを呼び出す方法

[2] 同一プロセス(クエリ)内から
ユーザ定義関数(UDF)を呼び出す方法

- I/Oパイプをオープン
- 標準入力でデータ受取
- シリアライズ/デシリアライズ処理
- 標準出力で処理結果をStreaming APIに受渡し

本検証では別プロセスよりも同一プロセスにデータ処理をさせるほうが約5．7倍高速に処理

データ処理の実装には、Hive Streaming機能よりもUDFを使用すべき

4-2 まとめ

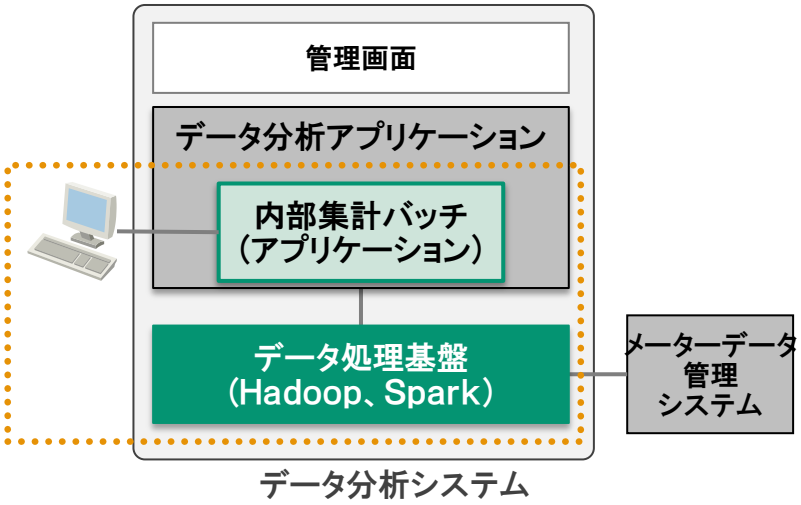
HITACHI
Inspire the Next

◆ HiveとSpark SQLの性能検証

実施内容

電力消費量データから配電設備の運用状況を把握するためのデータ分析システムを構築

分析処理負荷軽減のために基盤内部でメーターデータを予め集計する処理の性能を検証



性能検証の結果

Spark SQL (Spark Core)を用いると、目標時間(30分)以内で集計処理を完了できた

性能検証で得られた気付き

データソースのファイル形式にORCFileを用いるとCSVよりも性能が向上

- 特に特定列のみを読み込む処理で高い性能を発揮

全般的にHiveよりもSparkが高速に処理完了

- 特に途中の処理回数が多い場合に有利

他社商品名、商標等の引用に関する表示

HITACHI
Inspire the Next

- Hadoop、HiveおよびSparkは、Apache Software Foundationの米国およびその他の国における登録商標または商標です。
- その他記載の会社名、製品名などは、それぞれの会社の商標もしくは登録商標です。

HITACHI
Inspire the Next

END

Hadoop & Spark性能検証
～HiveとSpark SQLによる集計処理の比較～

2016/9/1

株式会社 日立製作所　OSSソリューションセンタ

木下翔伍

HITACHI

Inspire the Next