Oracle
DBA & Developer Days 2011

日本オラクル、今年最大の技術トレーニングイベント

2011年11月9日(水)～11月11日(金) シェラトン都ホテル東京

ORACLE

実はシンプル！RACチューニングの考え方

日本オラクル株式会社　基盤技術部
シニアエンジニア　佐々木亨

以下の事項は、弊社の一般的な製品の方向性に関する概要を説明するものです。また、情報提供を唯一の目的とするものであり、いかなる契約にも組み込むことはできません。以下の事項は、マテリアルやコード、機能を提供することをコミットメント（確約）するものではないため、購買決定を行う際の判断材料になさらないで下さい。オラクル製品に関して記載されている機能の開発、リリースおよび時期については、弊社の裁量により決定されます。

RAC に対するネガティブなイメージ

Cache Fusion が起こって

グローバル・キャッシュ待機イベント が発生

チューニングは
RAC 特有のテクニックが必要で複雑 なので

簡単には スケールしない

RAC に対して持っていただきたいイメージ

Cache Fusion が起こって

グローバル・キャッシュ待機イベント が発生しても

チューニングは
シングル・インスタンスと変わらない ので

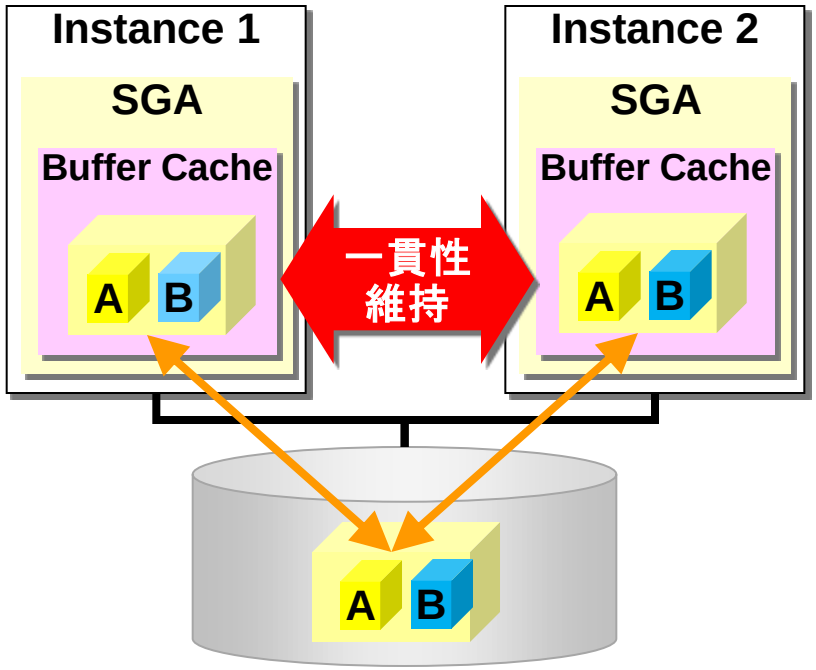
同じ手法を用いれば スケールする

Agenda

1. **はじめに**
2. RAC における考慮ポイントとチューニング例
3. スケール・アウトの例
4. まとめ

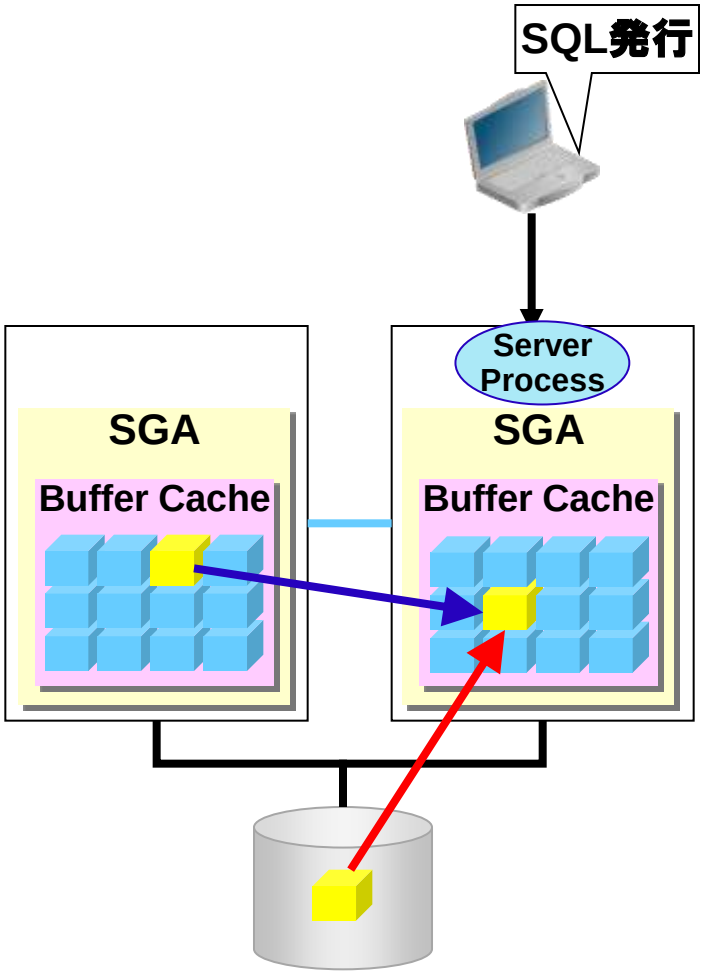
はじめに
RAC とは

- 共有ディスク/共有キャッシュ型のクラスタ・データベース
  - 全ノードが全データに直接アクセス可能
  - 複数ノード間でデータの一貫性を Cache Fusionの仕組みによって、自動で維持する
- シングルインスタンスと同じ構造
  - REDOログ/ UNDO表領域をインスタンス毎に追加する
- 障害ノードを切り離す
  - 全自動でリカバリ処理
  - 正常ノードで負荷分散



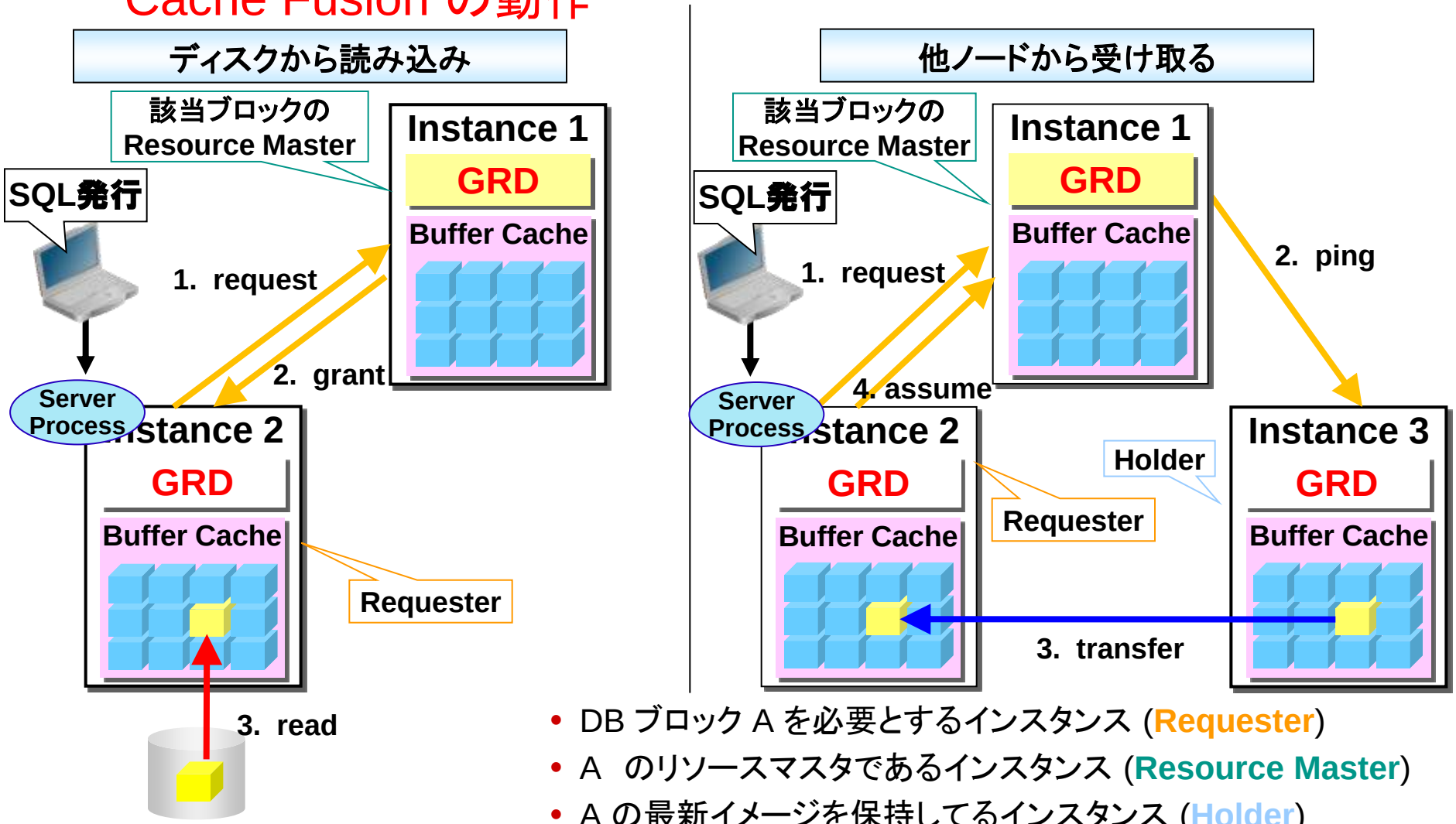
はじめに
RACのデータアクセス

- キャッシュ・ヒットした場合
  - キャッシュのデータを使用
- キャッシュ・ミスした場合
  - シングルインスタンスの場合
    - ディスクから読み込む
  - RACの場合
    - 他ノードのキャッシュから受け取る
    - ディスクから読み込む



はじめに

Cache Fusion の動作



- DB ブロック A を必要とするインスタンス (Requester)
- A のリソースマスタであるインスタンス (Resource Master)
- A の最新イメージを保持してるインスタンス (Holder)

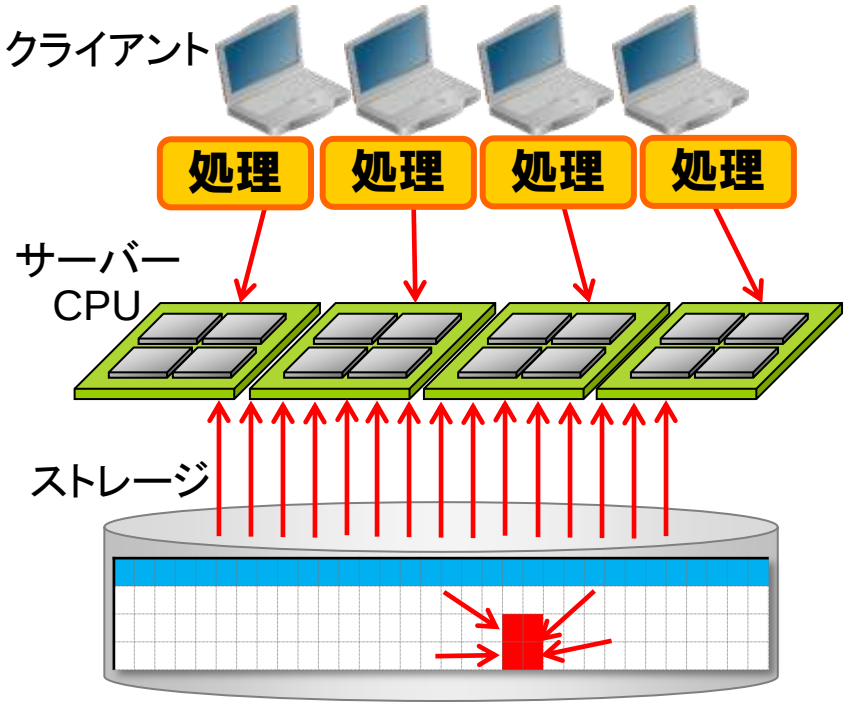
Agenda

1. はじめに
2. **RAC における考慮ポイントとチューニング例**
3. スケール・アウトの例
4. まとめ

CPU追加による同時実行性向上

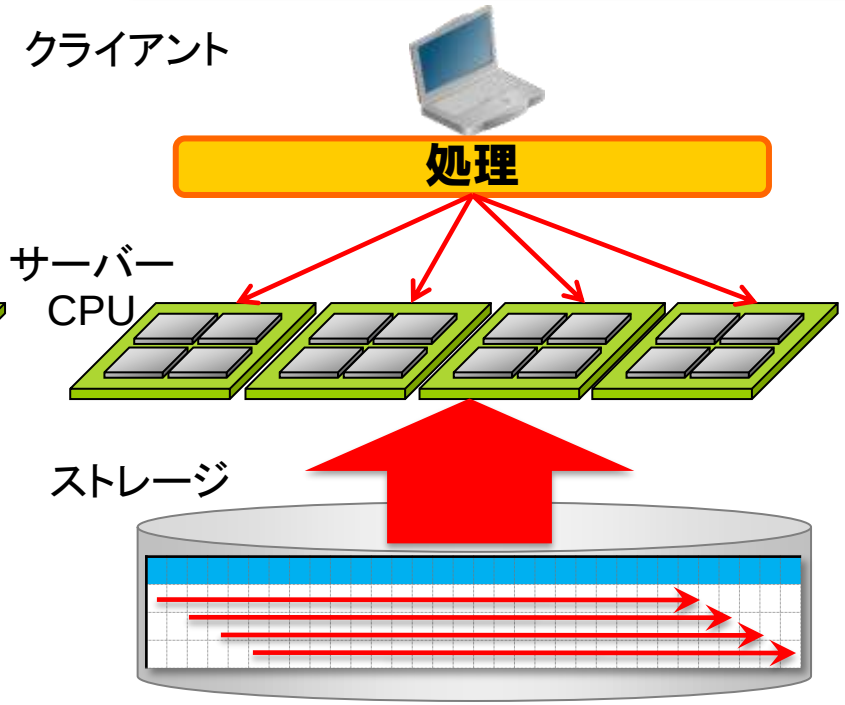
目的と課題

複数の処理を同時実行
（スループットを向上）



競合する個所（排他制御など）
がボトルネックになりがち

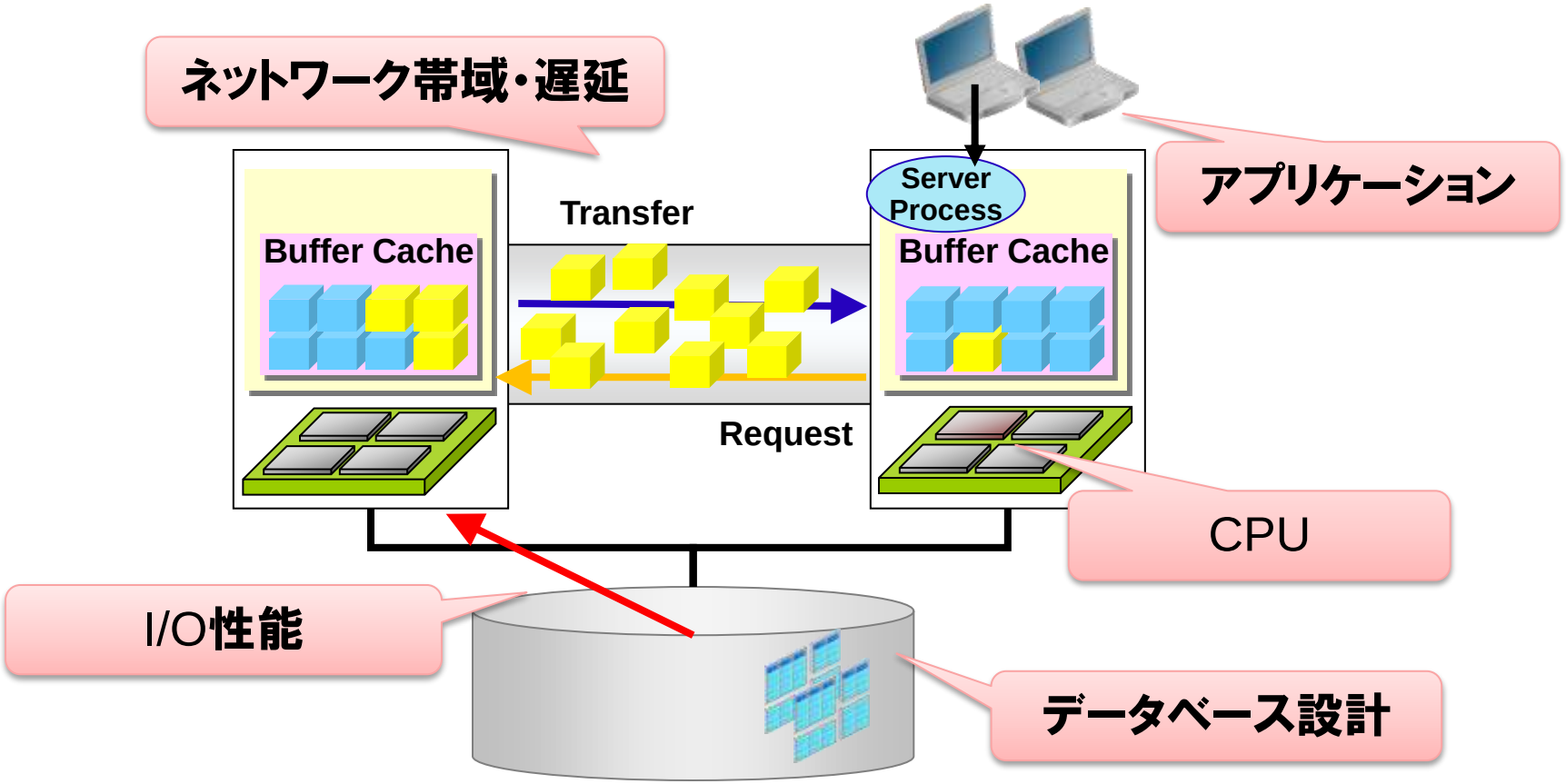
1つの処理を並列実行
（レスポンスタイムを向上）



大量のデータをCPUに供給する部分
がボトルネックになりがち

RACにおける考慮ポイント

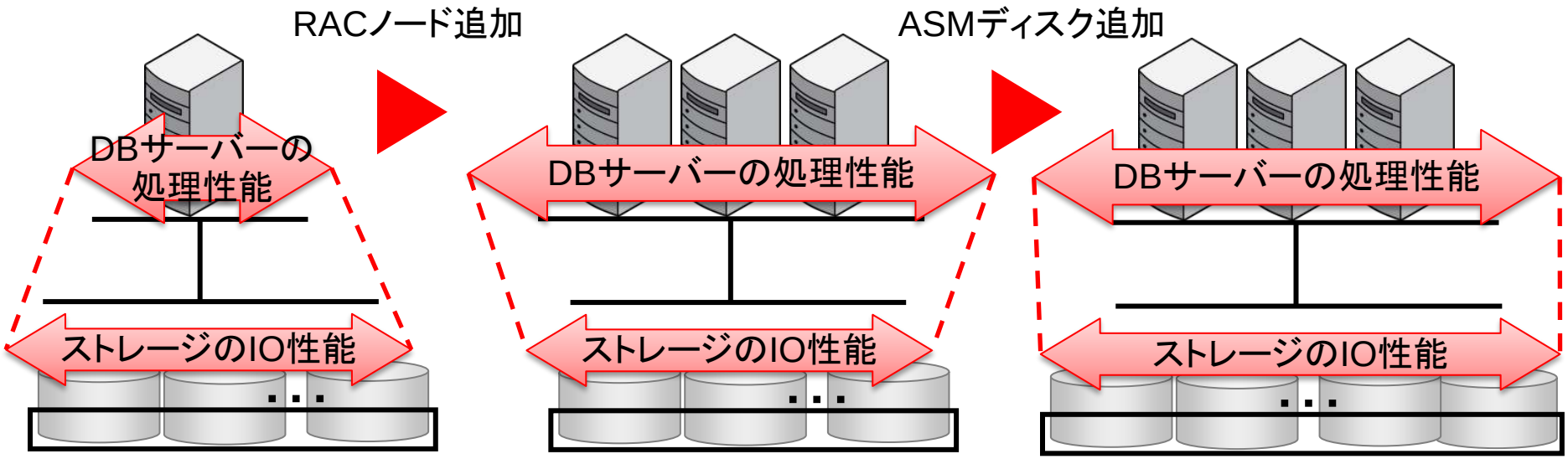
リソース競合、大量データを扱うという点からの考慮ポイント



RACにおける考慮ポイント

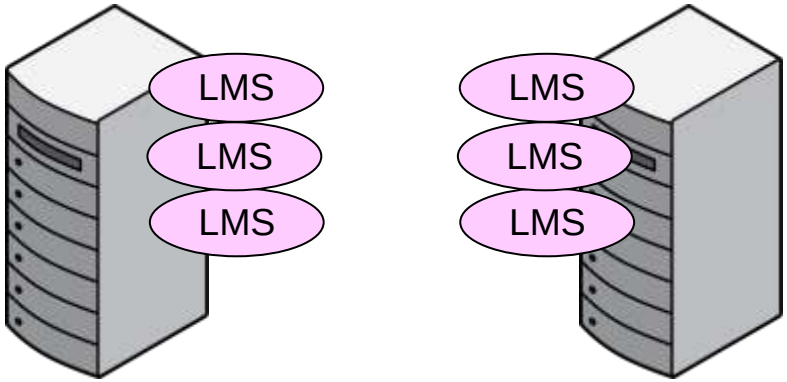
I/O性能

- ストレージはRACの全ノードで共有されるので、全体のI/O性能が重要となる
  - RAC でCPUを追加して処理能力を向上
  - ASM でストレージを追加してI/O性能を向上



RACにおける考慮ポイント
CPU

- マルチコア化が進んでいる中、CPU がボトルネックとなるケースは少ない
- 通信を行う LMS プロセスはCPU数をもとに複数起動され、CPUスケジューリングの優先度が高くなっているため、LMSが処理のボトルネックとなるケースは少ない



RACにおける考慮ポイント

ネットワーク帯域・遅延

- 遅延
  - ネットワーク上の伝搬にかかる時間(数百マイクロ秒)は、転送全体にかかる時間の数パーセントにも満たない
- 帯域
  - 使用するアプリケーション、CPU性能に依存する
  - 一般的にOLTPシステムでは1GbEで十分
    - 8KBのデータブロックを1秒間で10000回転送して1GbEが飽和
  - DWHシステムでは、1GbE以上の帯域が必要なケースもある

- 必要に応じて、NIC Bonding(負荷分散)、10GbE や Infiniband などの技術の使用を検討

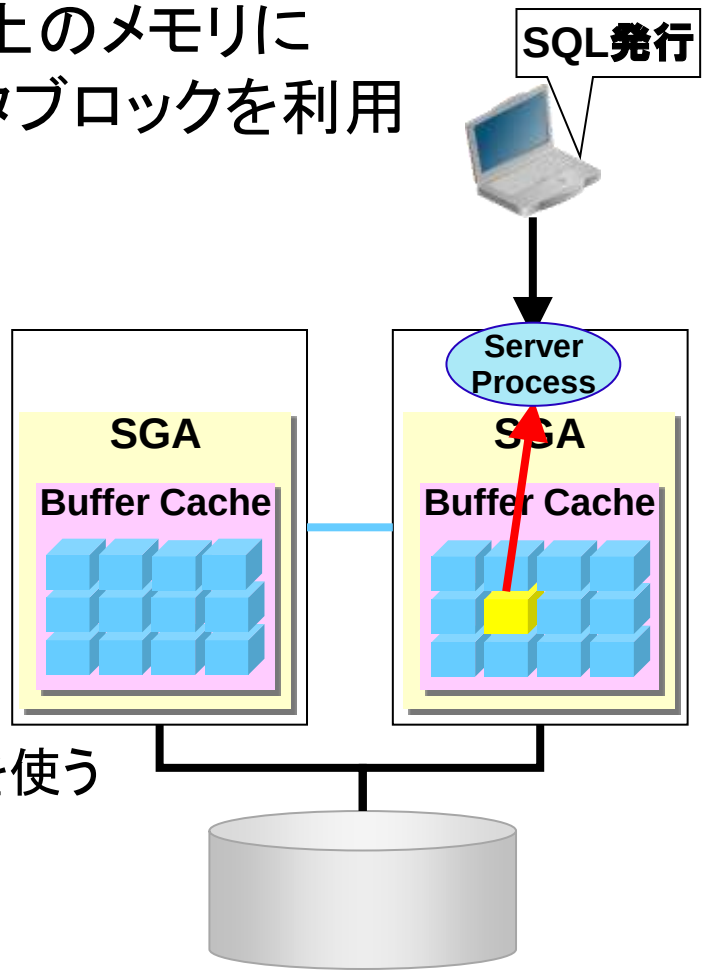
RACにおける考慮ポイント
アプリケーション・データベース設計

- アプリケーションやデータベース設計はチューニングにおいて最も考慮すべき点
  - データベースやOSのパラメータ・チューニングよりもはるかに効果が高い
  - シングル・インスタンスのチューニング・ポイントはRACでも同じように注目すべき

- チューニング・ポイントの例
  1. キャッシュ・ヒット率を改善
  2. ブロックへのアクセスを分散させる
  3. 余分な処理を起こさない
  4. SQLを並列化して大量データをうまく扱う

1. キャッシュ・ヒット率を改善

- SQLが実行されるローカル・ノード上のメモリに必要なデータがあれば、そのデータブロックを利用
  - メモリ(高速)へのアクセスで完結
  - Disk I/O(低速)は起こらない
- まずは、データベース全体でのキャッシュ・ヒット率を改善する
  - OLTP システムにおける Full Scan を避ける
  - Buffer Cache を適切なサイズにする
  - Oracle Database のデータ圧縮機能を使う
- 上記のチューニングは、シングル・インスタンスと同じ



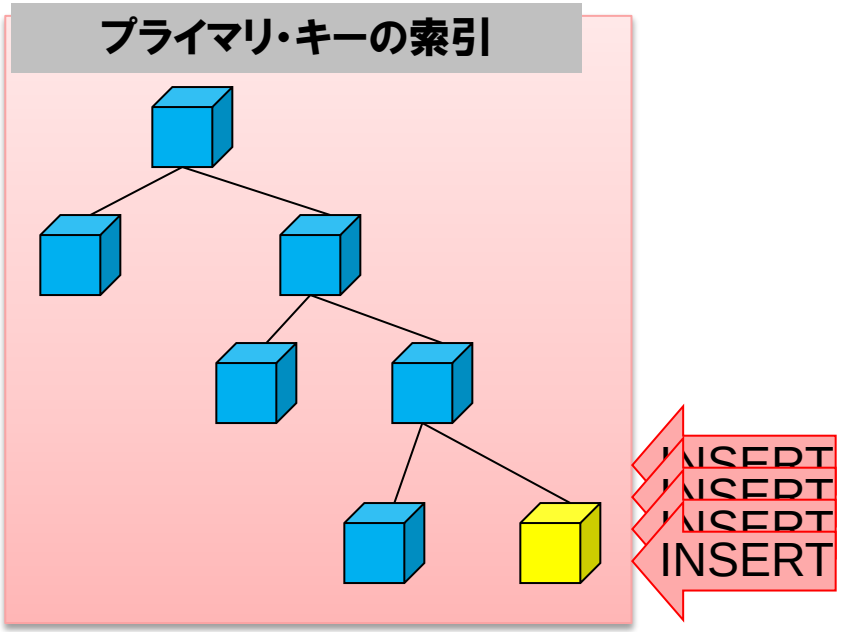
2. ブロックへのアクセスを分散させる

同一索引ブロックへのアクセスが集中する悪い例

Right Growing Index

- シーケンスから取得したような、単調増加する一意の値をINSERTする処理 (例:「注文番号」を INSERT)
- プライマリ・キーの索引の特定のリーフブロックに更新が集中する状況
- 同時実行数が増えればRAC に限らず発生する、良くない状況

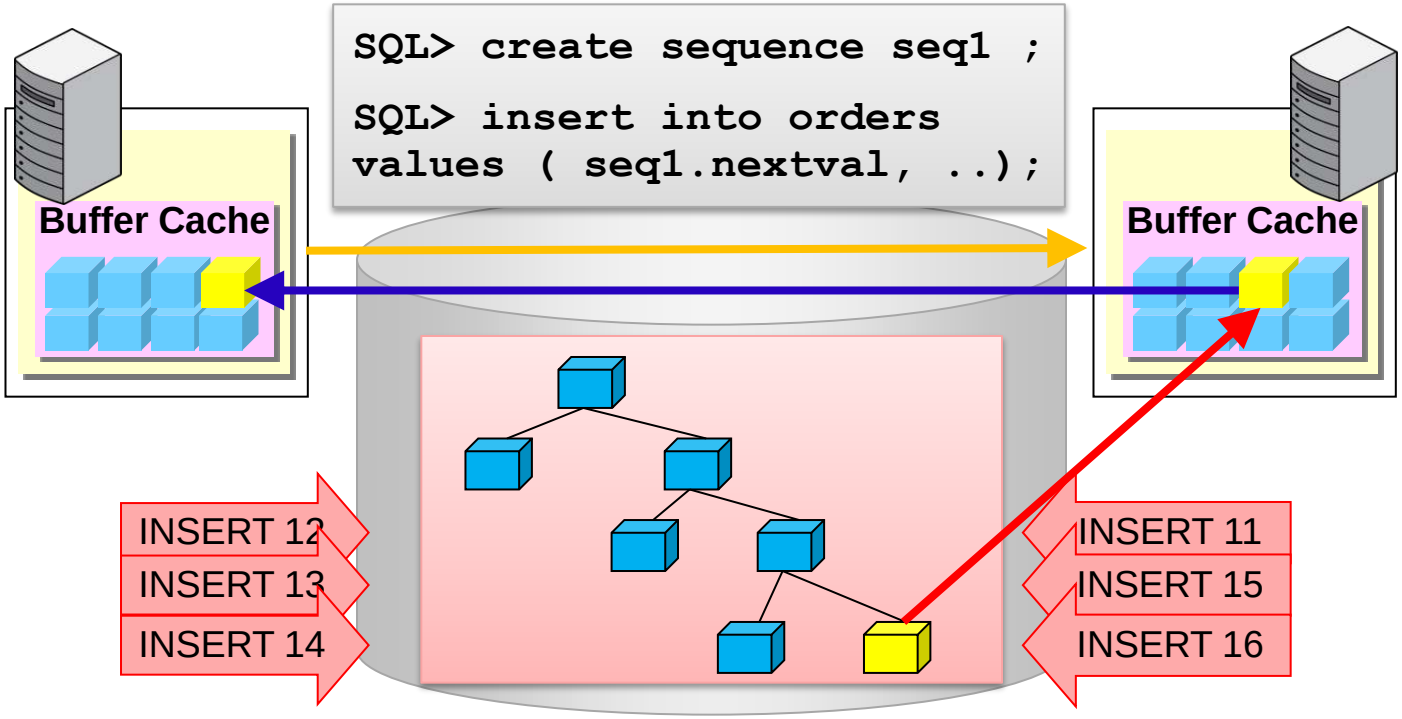
```
SQL> create table orders (
        orderid          number,
        customerid       number,
        orderdate        date,
                 :
        constraint orders_pk
        primary key (orderid)
     )

SQL> insert into orders values
     ( seq1.nextval,xxx,xxxxx,);

SQL> commit;
```



2. ブロックへのアクセスを分散させる

同一索引ブロックへのアクセスが集中する悪い例

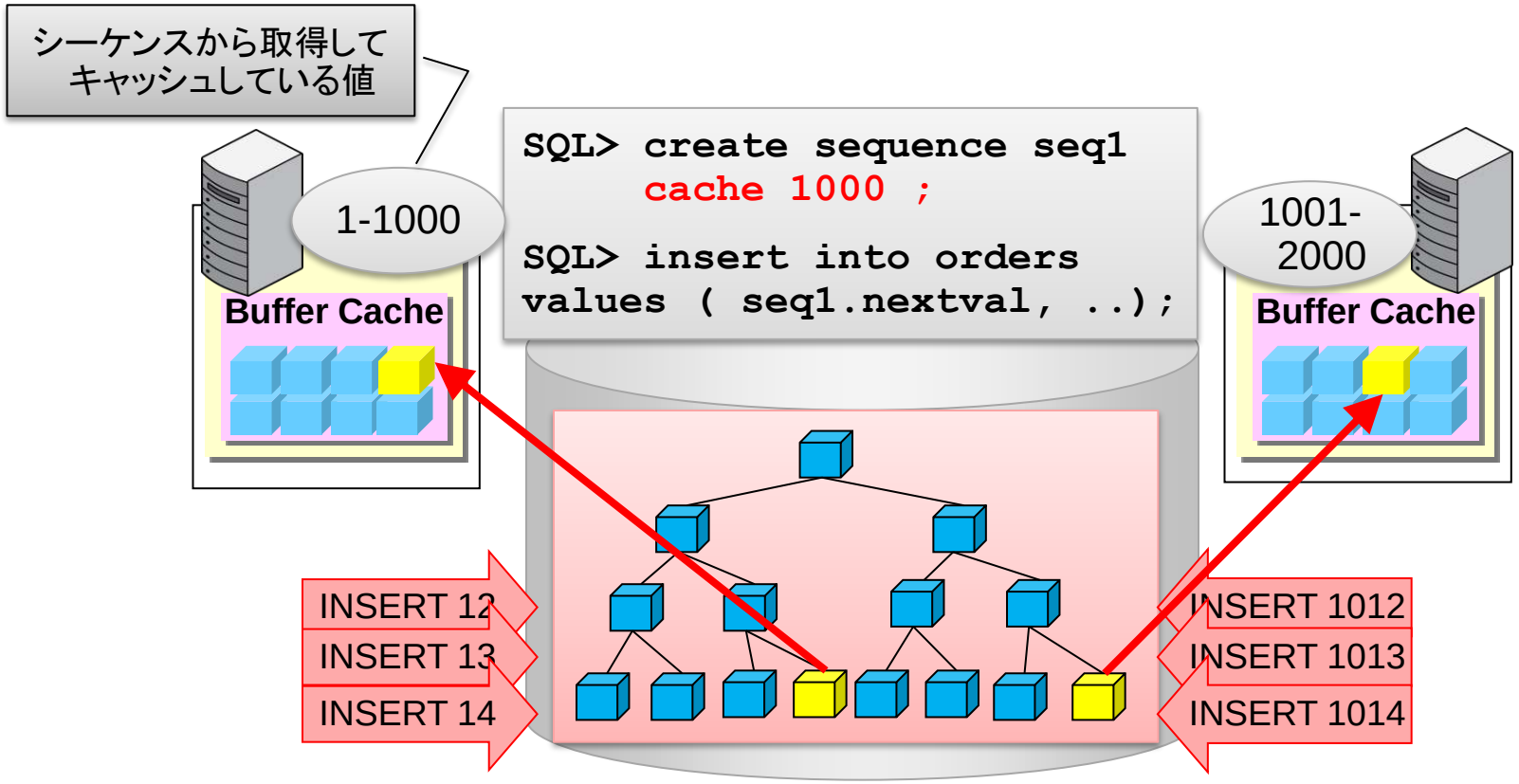
- 特定のリーフ・ブロックのノード間の転送が頻発する
- 他ノードのブロックに対する処理・転送を待つというように処理がシリアライズ化される部分が出てくる



2. ブロックへのアクセスを分散させる

シーケンスのキャッシュを増やす

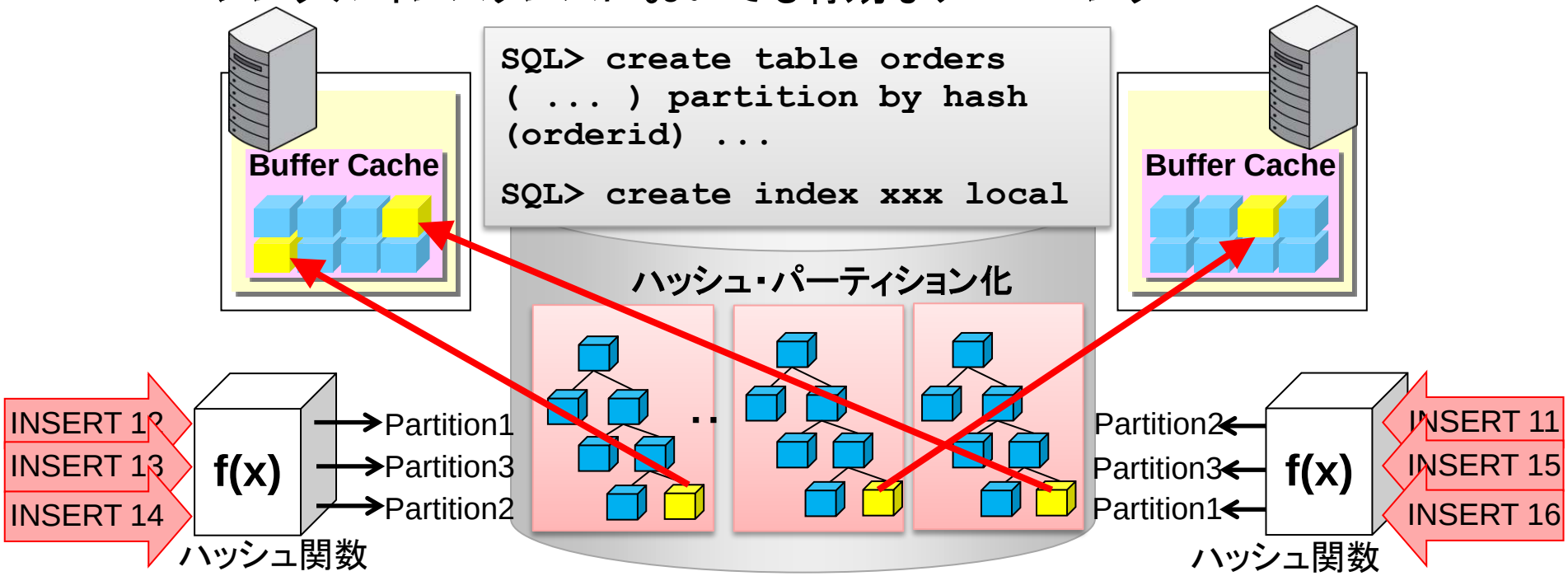
- シーケンスのキャッシュを増やすことで、各ノードからのINSERT処理が別のリーフ・ブロックに分散される



2. ブロックへのアクセスを分散させる

パーティショニング

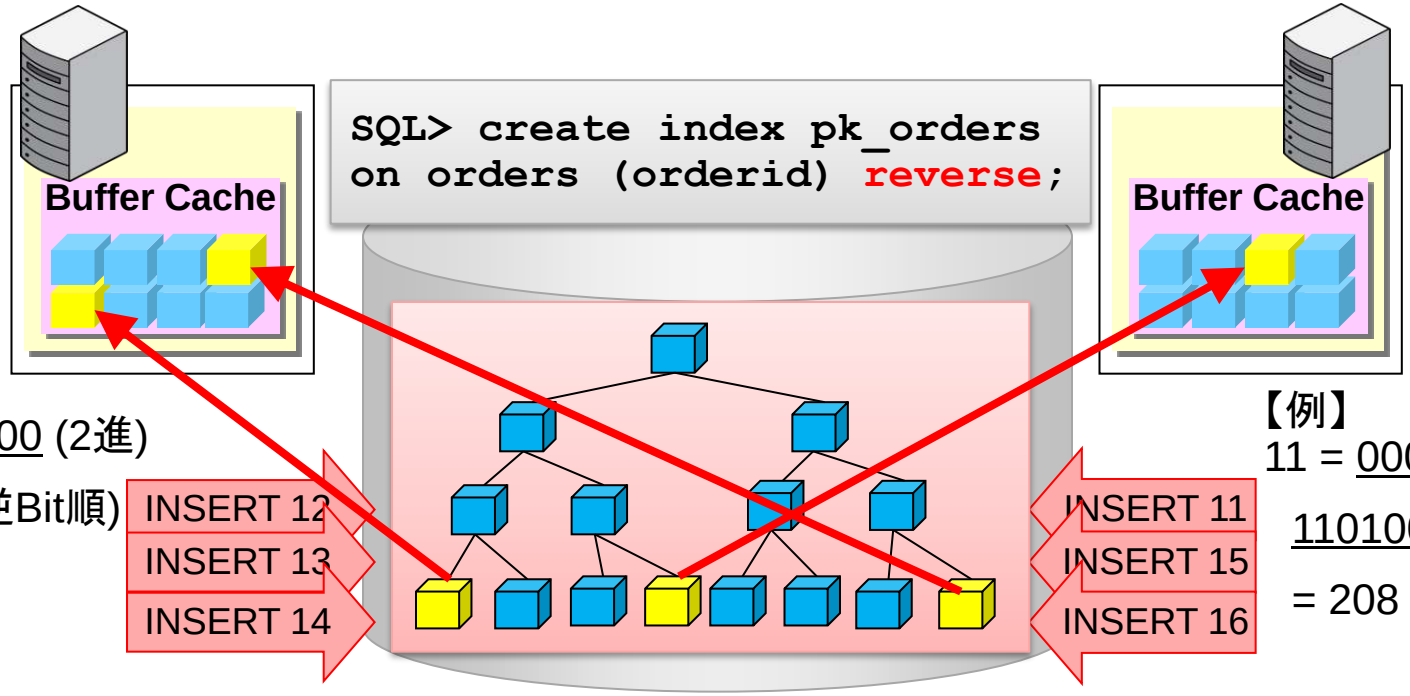
- プライマリ・キーの索引をパーティション化
  - テーブルをハッシュ・パーティション化してローカル索引を作成
- アクセス対象のリーフ・ブロックを全体で分散可能
  - シングルインスタンスにおいても有効なチューニング



2. ブロックへのアクセスを分散させる

逆キー索引

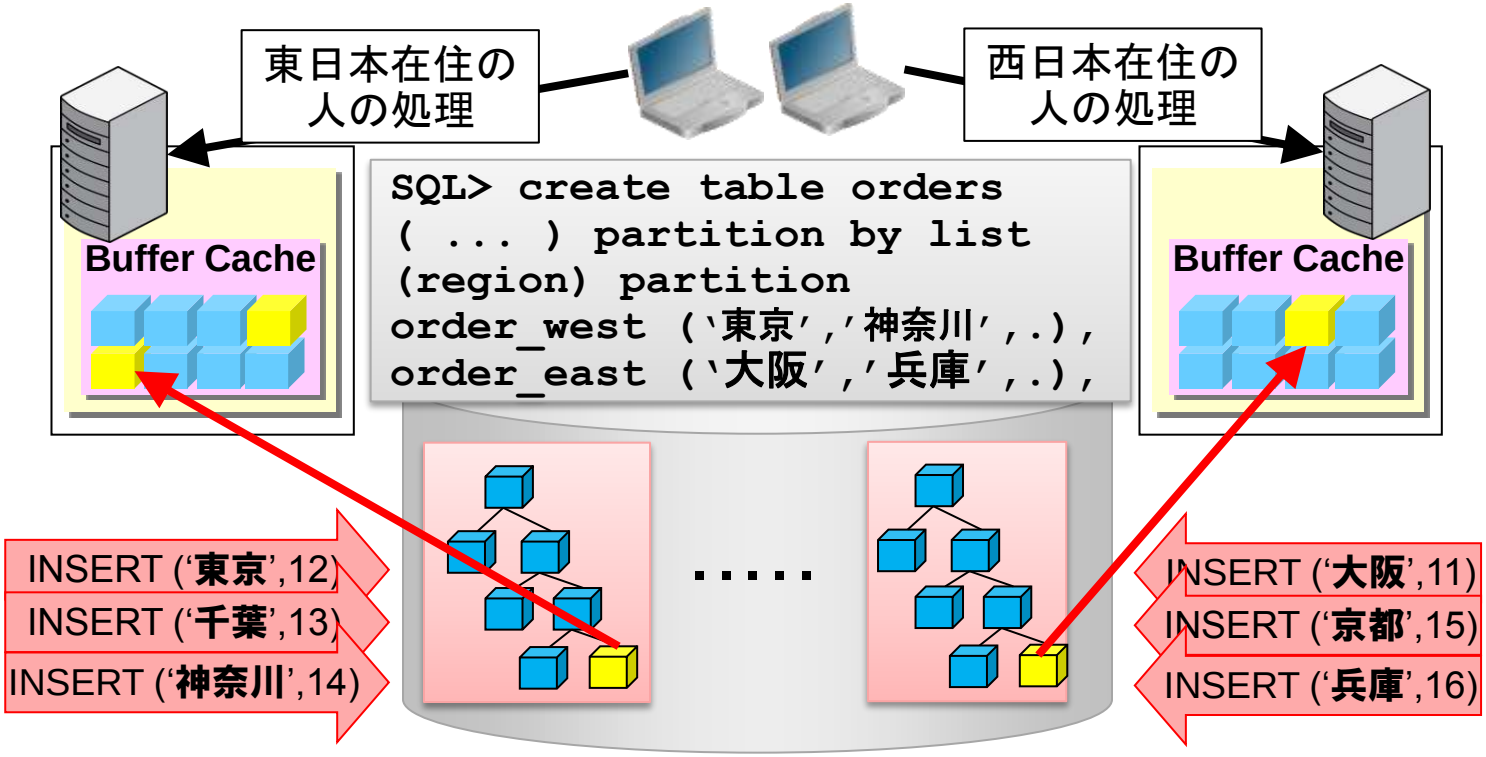
- プライマリ・キーの索引を逆キー索引化(逆ビット順に格納)
  - 大小関係が変化し、連続したキーでも異なるブロックに挿入され易くなる
  - シングルインスタンスでも有効なチューニング
  - 索引を使った範囲検索ができなくなる点には注意



2. ブロックへのアクセスを分散させる

アプリケーション・パーティショニング

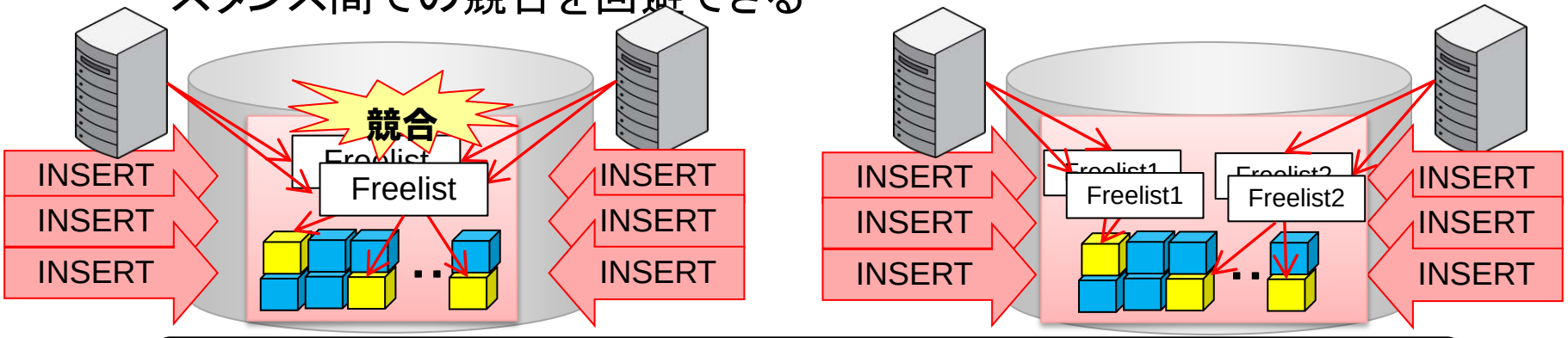
- 異なるノードから別のブロックにアクセスするようアプリケーションで制御(アプリケーション・ロジック的に可能な場合のみ)
- レンジ、リストパーティションなどと組合せると効果的



2. ブロックへのアクセスを分散させる

セグメント・ヘッダー・ブロックの競合

- Freelist
  - データ挿入時に探索する空きブロックのリンク・リスト
  - 多くのプロセスが同時に挿入処理を行うと、セグメント・ヘッダー・ブロック獲得のための競合が発生
- Freelist Group
  - インスタンス毎にFreelist 探索のブロックが割り当てられるため、インスタンス間での競合を回避できる

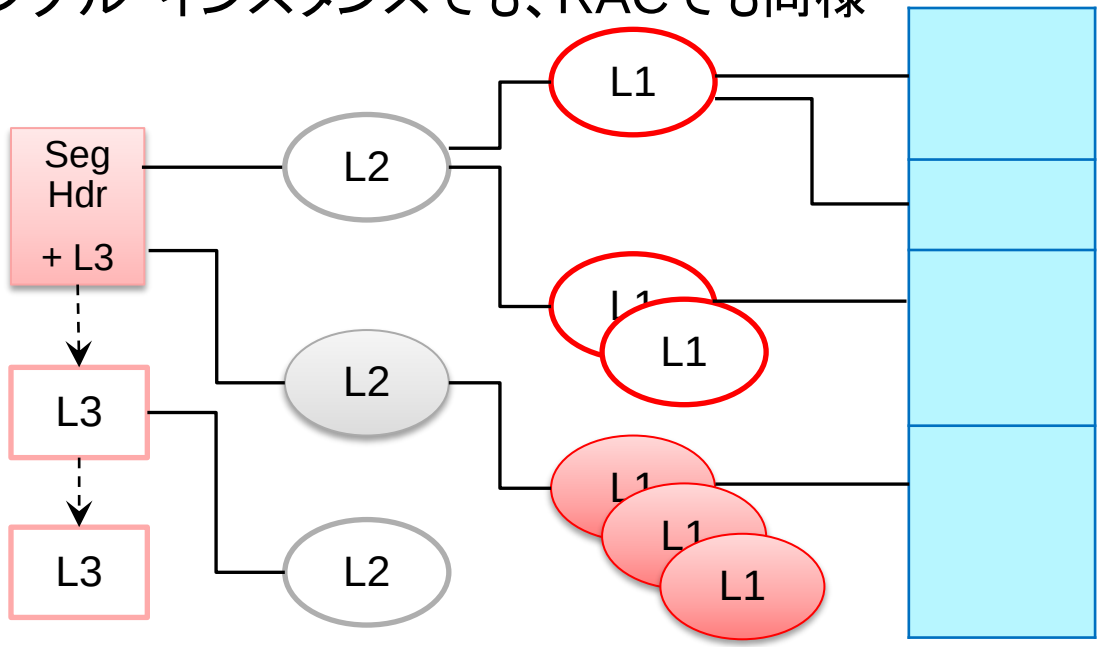


古いバージョンのソフトウェアだと上記のことまで考える必要がある

2. ブロックへのアクセスを分散させる

ASSM(Automatic Segment Space Management)

- 新しいバージョンのソフトウェアを使えば、ASSMの機能が使われるので前ページの内容を意識する必要がない
  - リスト構造ではなく、ツリー構造になっており、空きブロック探索時にセグメント・ヘッダー・ブロックの競合が起こりにくい
  - シングル・インスタンスでも、RACでも同様



3. 余分な処理を起こさない

不要なパースをさせないためバインド変数を使用

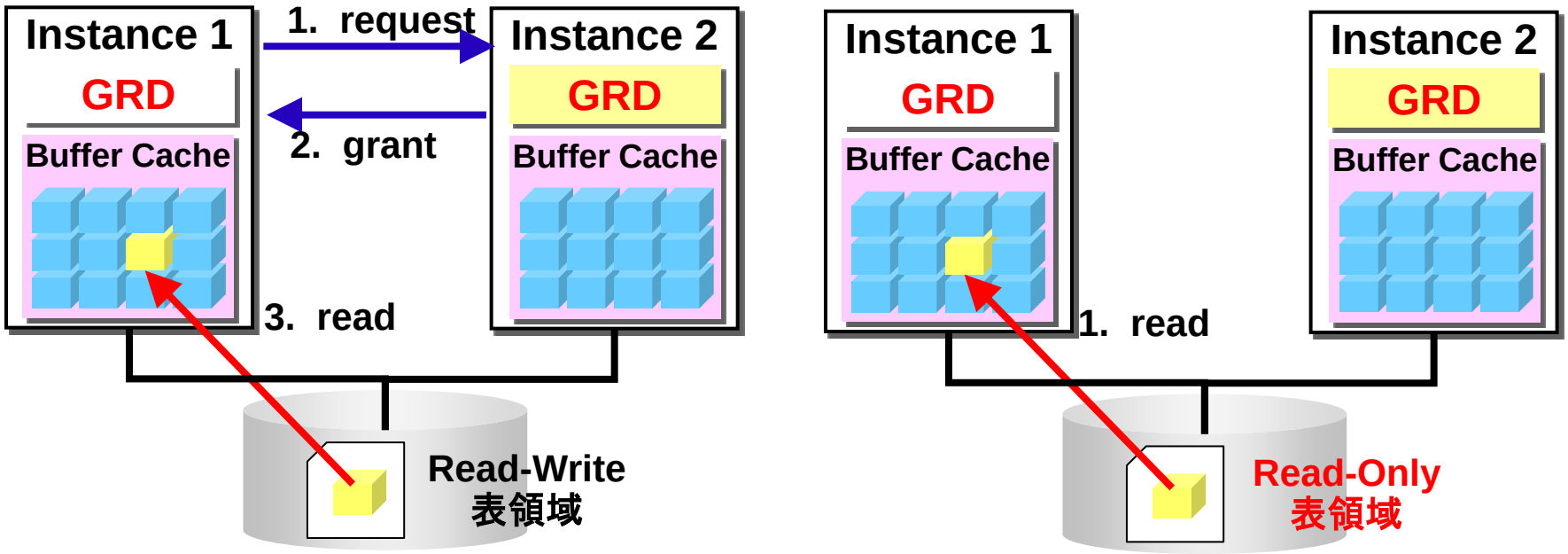
- RAC 環境では、Library Cache はノード間で処理される
  - インターコネクトを介した通信、オペレーションが入る

- 不要なパースを避けるというのはシングル・インスタンスでも言えること

3. 余分な処理を起こさない

読み取り専用のデータはRead-Only表領域に配置

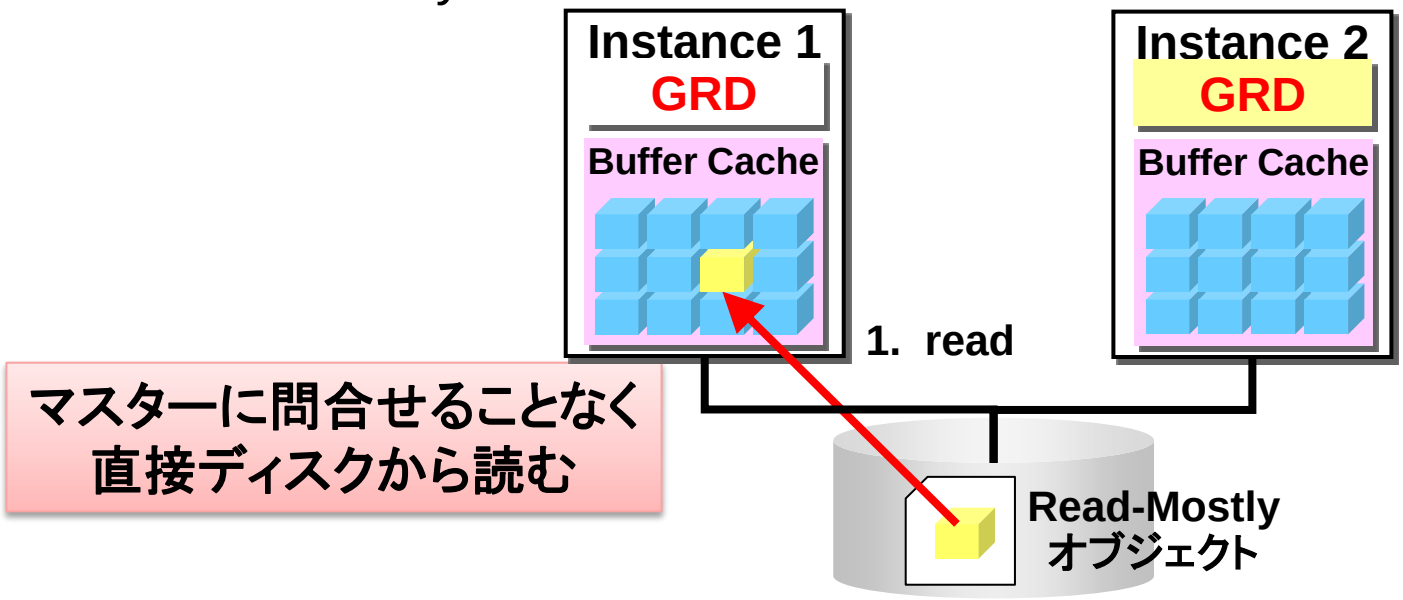
- 読み取り専用の表領域中のテーブルについては、Disk からの読み取りは Global Cache Operation を伴わない
  - 余分な通信によるオーバーヘッドを抑えることができる



3. 余分な処理を起こさない

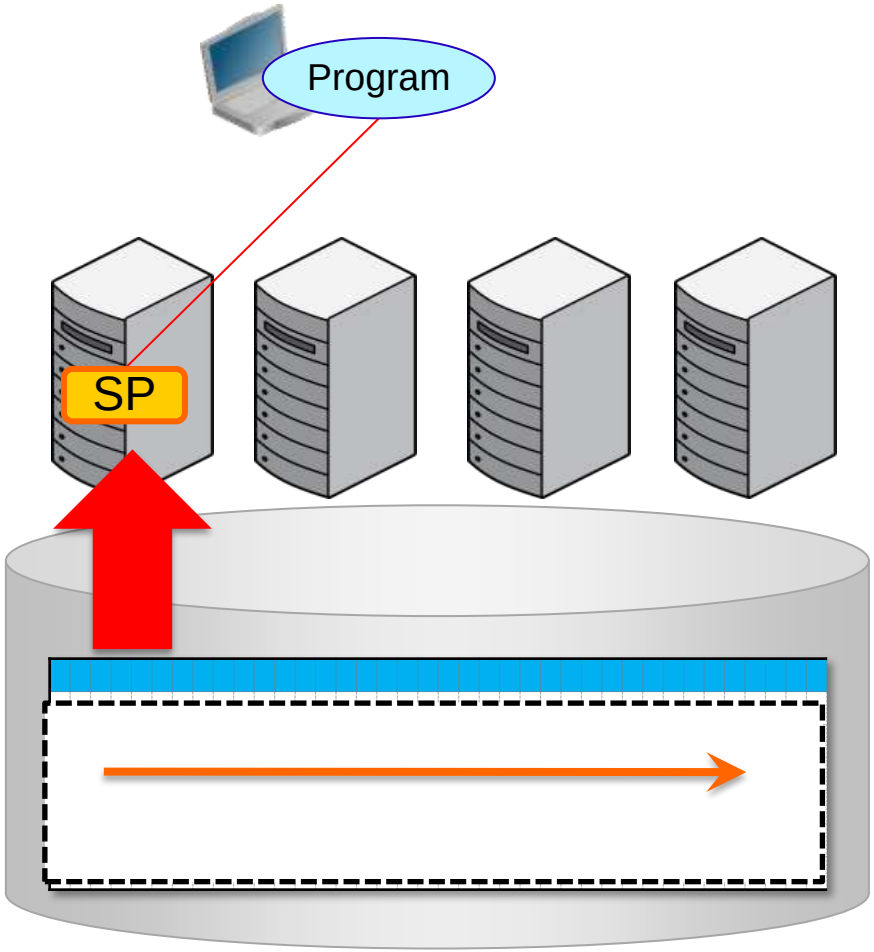
Read-Mostly Locking

- 99%はRead処理だけど、1%はWrite処理というオブジェクト (Read-Mostly) については、Read-Only にはできない
- Read-Mostly であるオブジェクトについては、マスターに問い合わせることなく直接ディスクから読み出す
  - Read-Only オブジェクトは自動で判断される



4. SQLを並列化して性能向上

並列化をしていない場合

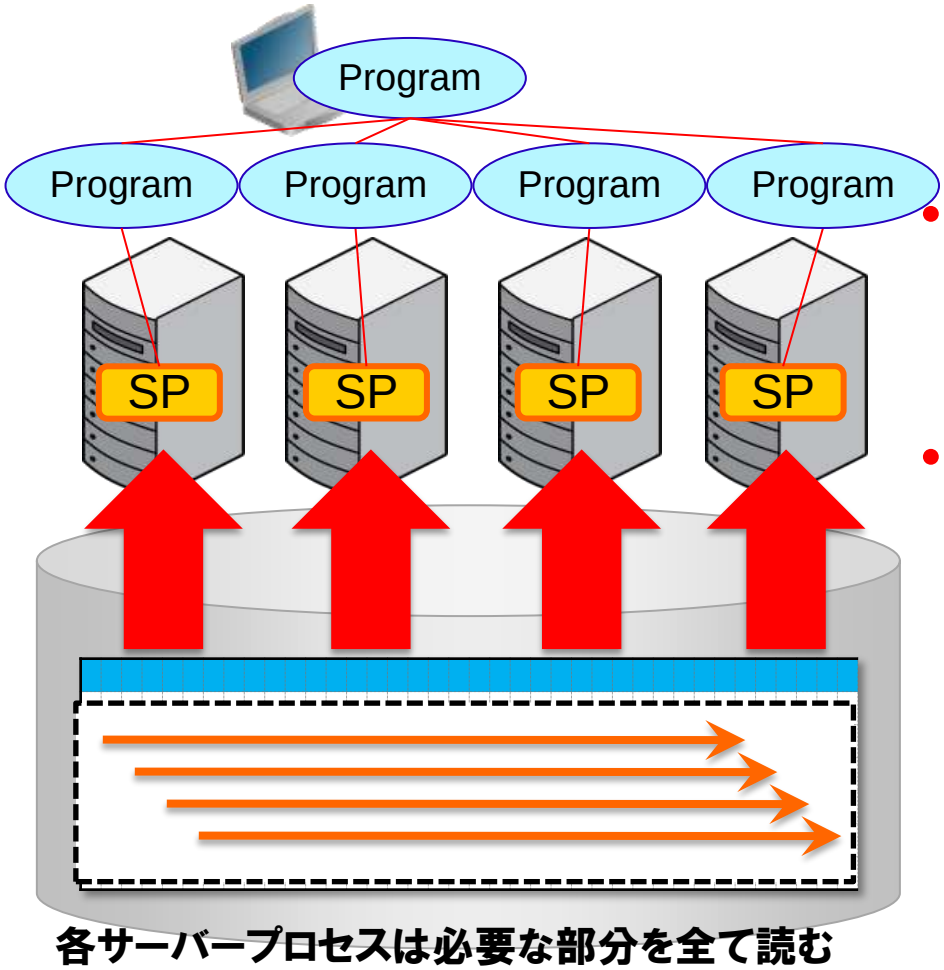


SP：サーバー・プロセス

- RACの他ノードのCPUにデータを供給できない

4. SQLを並列化して性能向上

プログラムで並列化する悪い例



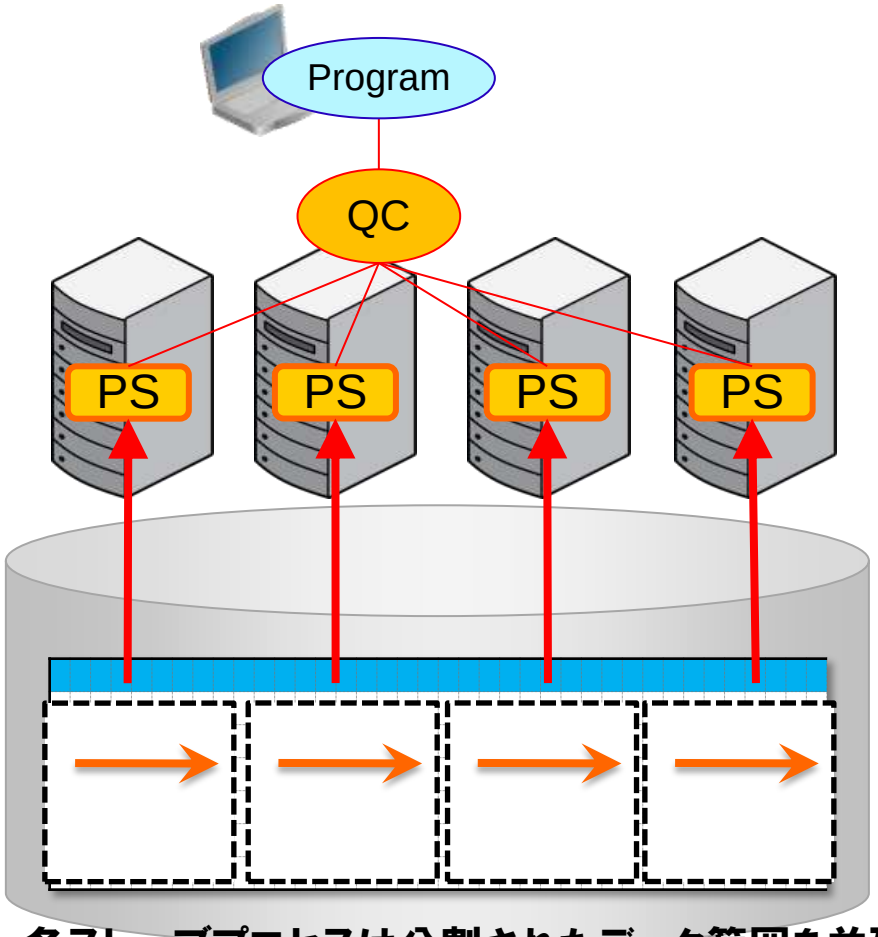
SP：サーバー・プロセス

- 同じ表に対してプログラムを分割した分だけFull Scan が同時に実行
- I/Oボトルネックとなり、並列度を挙げてもスケールしない

4. SQLを並列化して性能向上

Internode Parallel Query

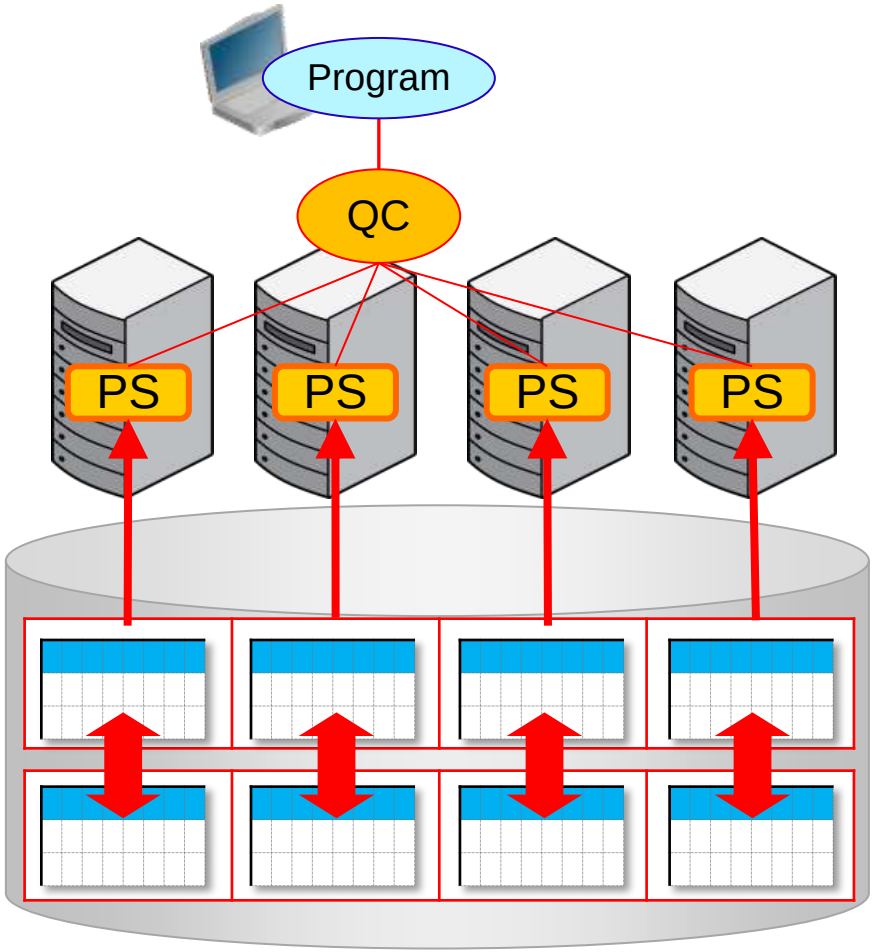
QC：クエリ・コーディネータ
PS：パラレル・スレーブ・プロセス



- 1つのSQLが複数ノードで並列化される
- 扱うデータは自動的に分割され、スキャン範囲の担当を動的に決定する
  - 各スレーブプロセスが異なるブロックを担当する
  - スレーブプロセスの実行時間が均等になるように

各スレーブプロセスは分割されたデータ範囲を並列実行

4. SQLを並列化して性能向上



QC：クエリ・コーディネータ
PS：パラレル・スレーブ・プロセス

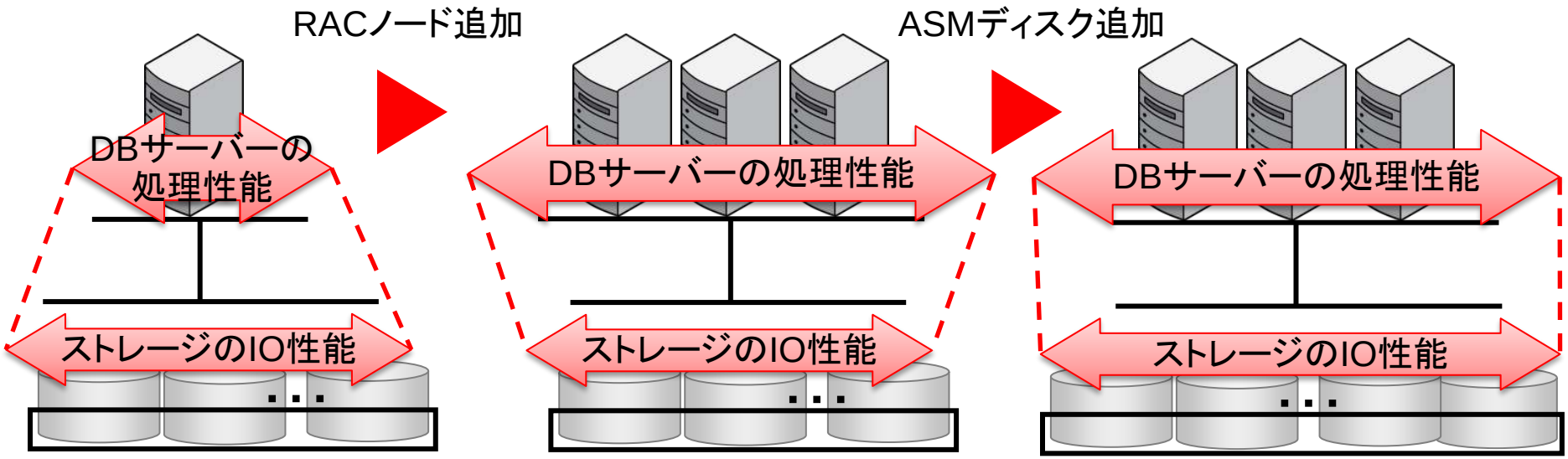
- パーティション表使用時は、更に賢くデータを分割
- 同じパーティション方式、かつパーティション・キー同士のJoin
- ノード毎にパーティションのJoinを割り当てる

上記のような処理によって
大量データ処理が可能となる

RACにおける考慮ポイント

I/O性能

- ストレージはRACの全ノードで共有されるので、全体のI/O性能が重要となる
  - RAC でCPUを追加して処理能力を向上
  - ASM でストレージを追加してI/O性能を向上



小まとめ

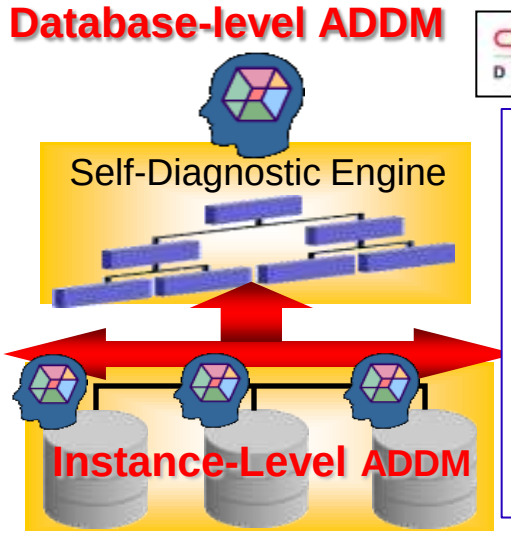
- シングルインスタンスで必要なチューニングはRACでも同じように必要
  1. キャッシュ・ヒット率を改善
  2. ブロックへのアクセスを分散させる
  3. 余分な処理を起こさない

- 競合を減らす、大量データを効率よく扱うための自動化機能が提供されている
  1. Read-Mostly Lock
  2. ASSM
  3. Internode Parallel Query

ADDM for RAC

概要

- RACデータベース全体のパフォーマンス診断・アドバイス
  - 各インスタンスの統計情報の累計をもとに診断
  - RAC固有のグローバル・リソース（共有ディスクへのI/Oやインターコネクトのトラフィック）の診断時に特に有効
- 1時間ごとのAWRスナップショット取得と同時に、インスタンスレベルとデータベースレベルでのADDMが実行



ORACLE 11g DATABASE

ADDMパフォーマンス分析

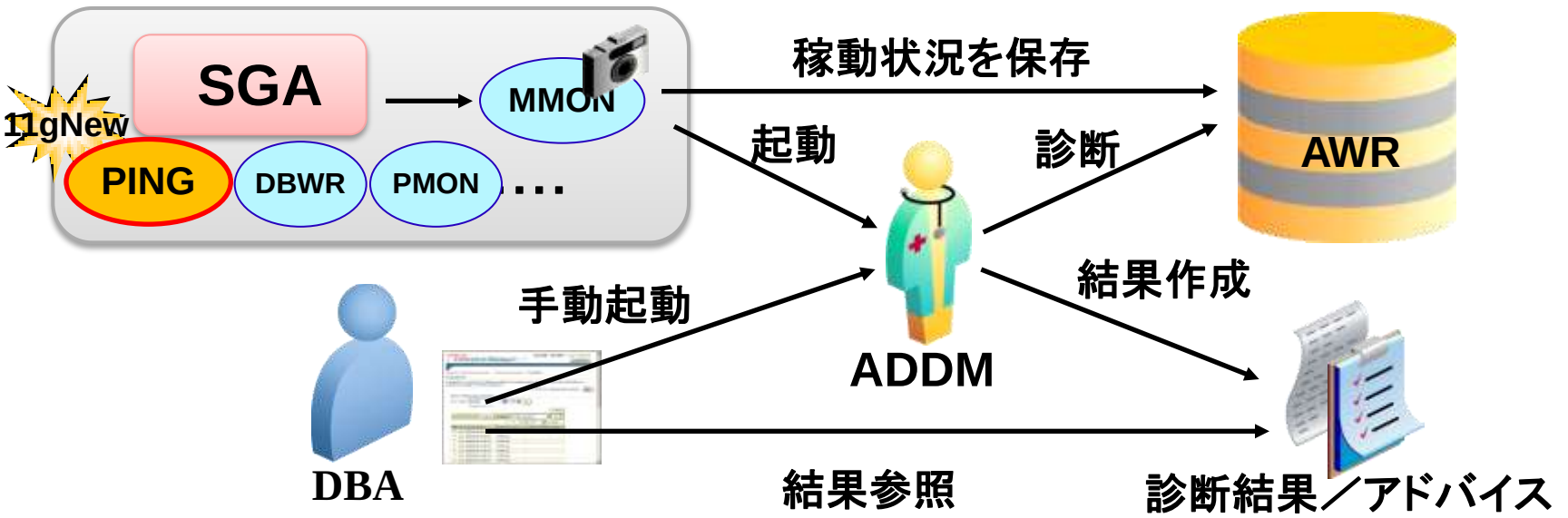
期間開始時間 **2007/08/14 16時15分21秒 JST**　持続期間(分) **8.67**　インスタンス **すべて**

影響(%) ▽	結果	影響を受けるインスタンス
66.2	Top SQL by DB Time	
52.3	Top SQL by "Cluster" Wait	
31.3	Interconnect Latency	2/2
29.7	Commits and Rollbacks	2/2
15.9	Interconnect Multiblock Requests	

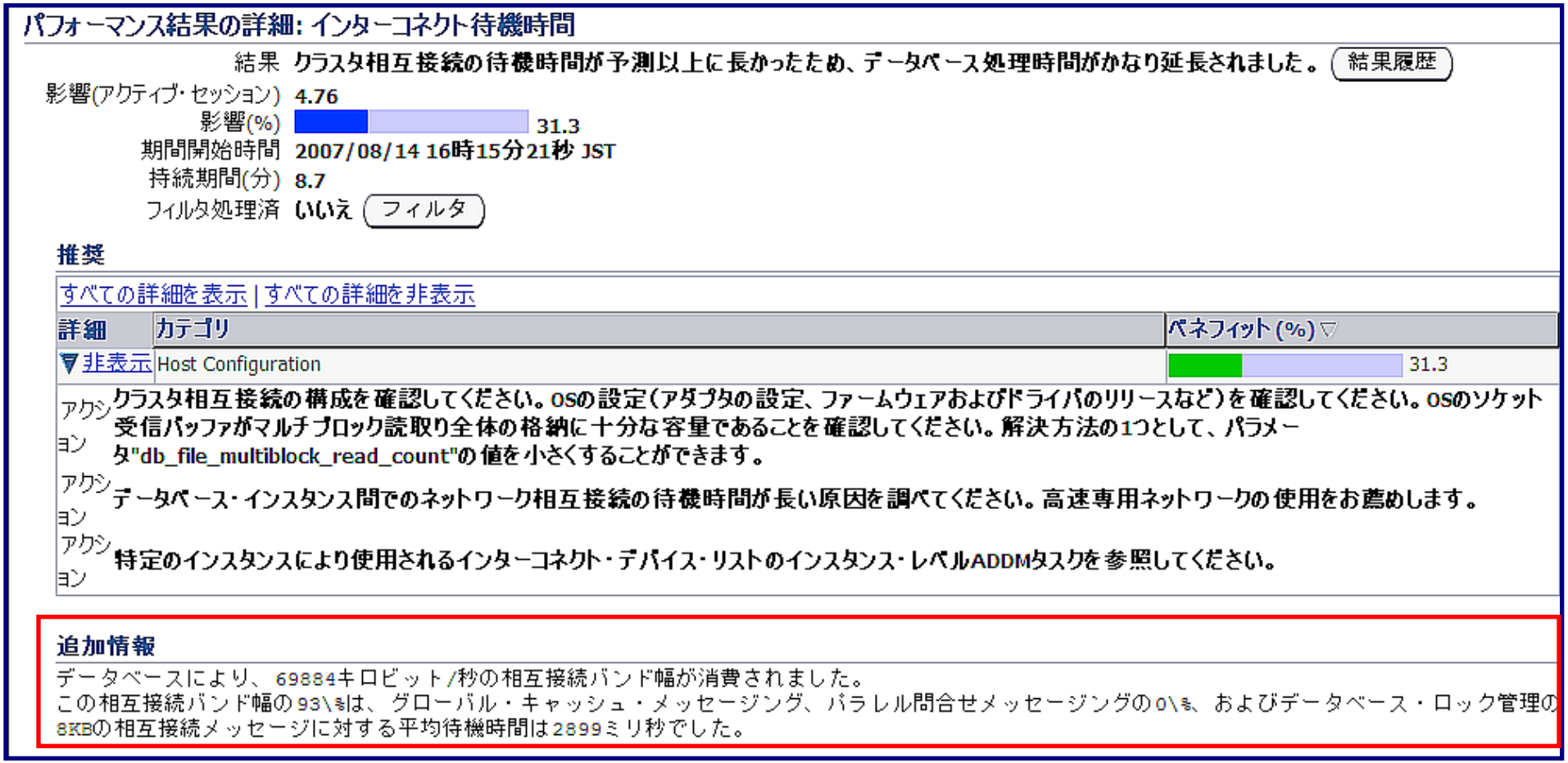
ADDM for RAC
アーキテクチャ

インターコネクトトラフィックに関する統計
- OSから見たネットワークデバイスの統計
- PING実行時のレスポンス時間
- インターコネクトトラフィック統計

追加



ADDM for RAC
インターコネクト待機時間



パフォーマンス結果の詳細: インターコネクト待機時間

結果 クラスタ相互接続の待機時間が予測以上に長かったため、データベース処理時間がかなり延長されました。 結果履歴

影響(アクティブ・セッション) 4.76

影響(%) 31.3

期間開始時間 2007/08/14 16時15分21秒 JST

持続期間(分) 8.7

フィルタ処理済 いいえ フィルタ

推奨

すべての詳細を表示 | すべての詳細を非表示

詳細	カテゴリ	ベネフィット (%)
非表示	Host Configuration	31.3

アクション クラスタ相互接続の構成を確認してください。OSの設定(アダプタの設定、ファームウェアおよびドライバのリリースなど)を確認してください。OSのソケット受信バッファがマルチブロック読取り全体の格納に十分な容量であることを確認してください。解決方法の1つとして、パラメータ"db_file_multiblock_read_count"の値を小さくすることができます。

アクション データベース・インスタンス間でのネットワーク相互接続の待機時間が長い原因を調べてください。高速専用ネットワークの使用をお薦めします。

アクション 特定のインスタンスにより使用されるインターコネクト・デバイス・リストのインスタンス・レベルADDMタスクを参照してください。

追加情報

データベースにより、69884キロビット/秒の相互接続バンド幅が消費されました。
この相互接続バンド幅の93%は、グローバル・キャッシュ・メッセージング、パラレル問合せメッセージングの0%、およびデータベース・ロック管理の8KBの相互接続メッセージに対する平均待機時間は2899ミリ秒でした。

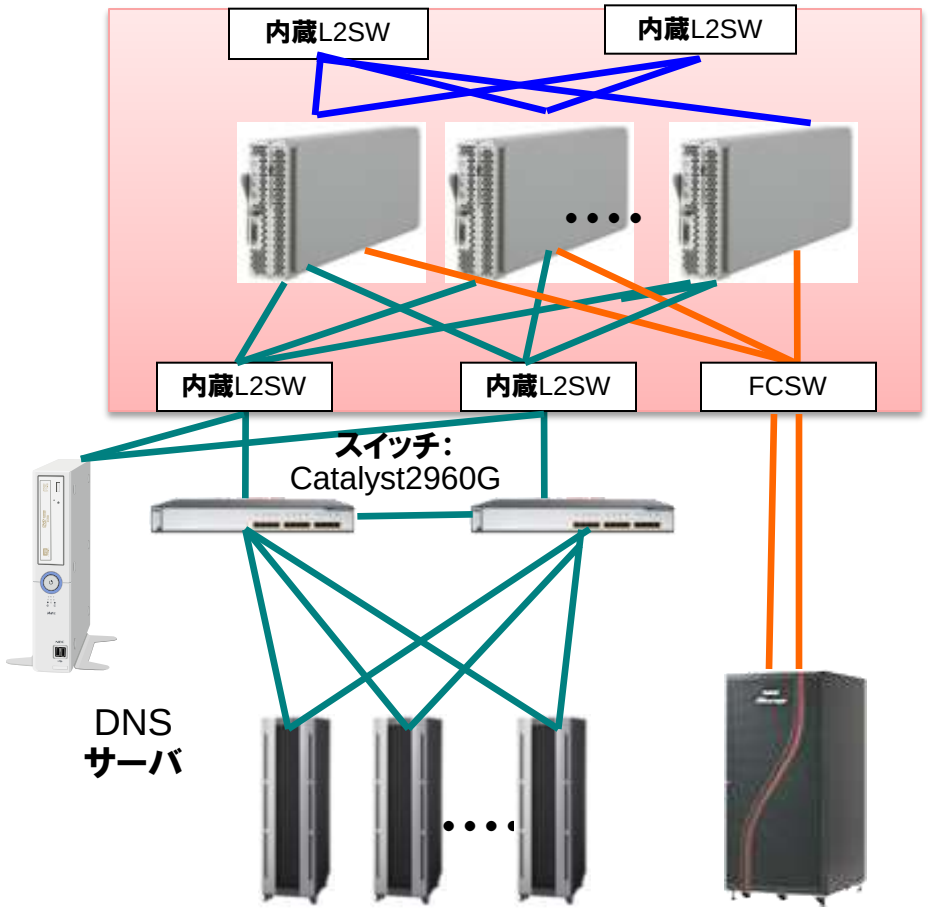
Agenda

1. はじめに
2. RAC における考慮ポイントとチューニング例
3. **スケール・アウトの例**
4. まとめ

スケール・アウトの例

ハードウェア構成

DBサーバ ×16台



DBサーバー（1台あたり）	**本体:** Express5800/B120a-d (N8400-089) **CPU:** インテル Xeon プロセッサー X5550　4Core * 2スレッド* 2CPU **メモリ:**48GB
クライアント（1台あたり）	**本体:** ECOCENTER（NE1000-001） **CPU:** 8Core **メモリ:**16GB
ストレージ	**本体:** iStorage S4900 **キャッシュメモリ:** 100GB

スケール・アウトの例
アプリケーション

- Webショッピングサイトを模したアプリケーションを想定

TX1（更新あり）

1. ユーザー・サインオン
   - SELECT ... FROM account, profile, signon, bannerdata ...
2. 商品検索
   - SELECT ... FROM category ...
   - SELECT ... FROM product ...
3. 商品選択
   - SELECT ... FROM item, product ...
4. 在庫数チェック
   - SELECT ... FROM inventory ...
5. 注文
   - (SELECT ordernum.nextval FROM dual)

更新処理

- INSERT INTO orders ...
- INSERT INTO orderstatus ...
- INSERT INTO lineitem ...
- UPDATE inventory ...
- COMMIT

TX2（検索のみ）

1. ユーザー・サインオン
   - SELECT ... FROM account, profile, signon, bannerdata ...
2. 商品検索
   - SELECT ... FROM category ...
   - SELECT ... FROM product ...
3. 商品選択
   - SELECT ... FROM item, product ...
4. 在庫数チェック
   - SELECT ... FROM inventory ...

TX1とTX2をそれぞれ「1トランザクション」とする

アプリケーションパーティショニングは実施しない

商品検索は平均100件程度がヒットする

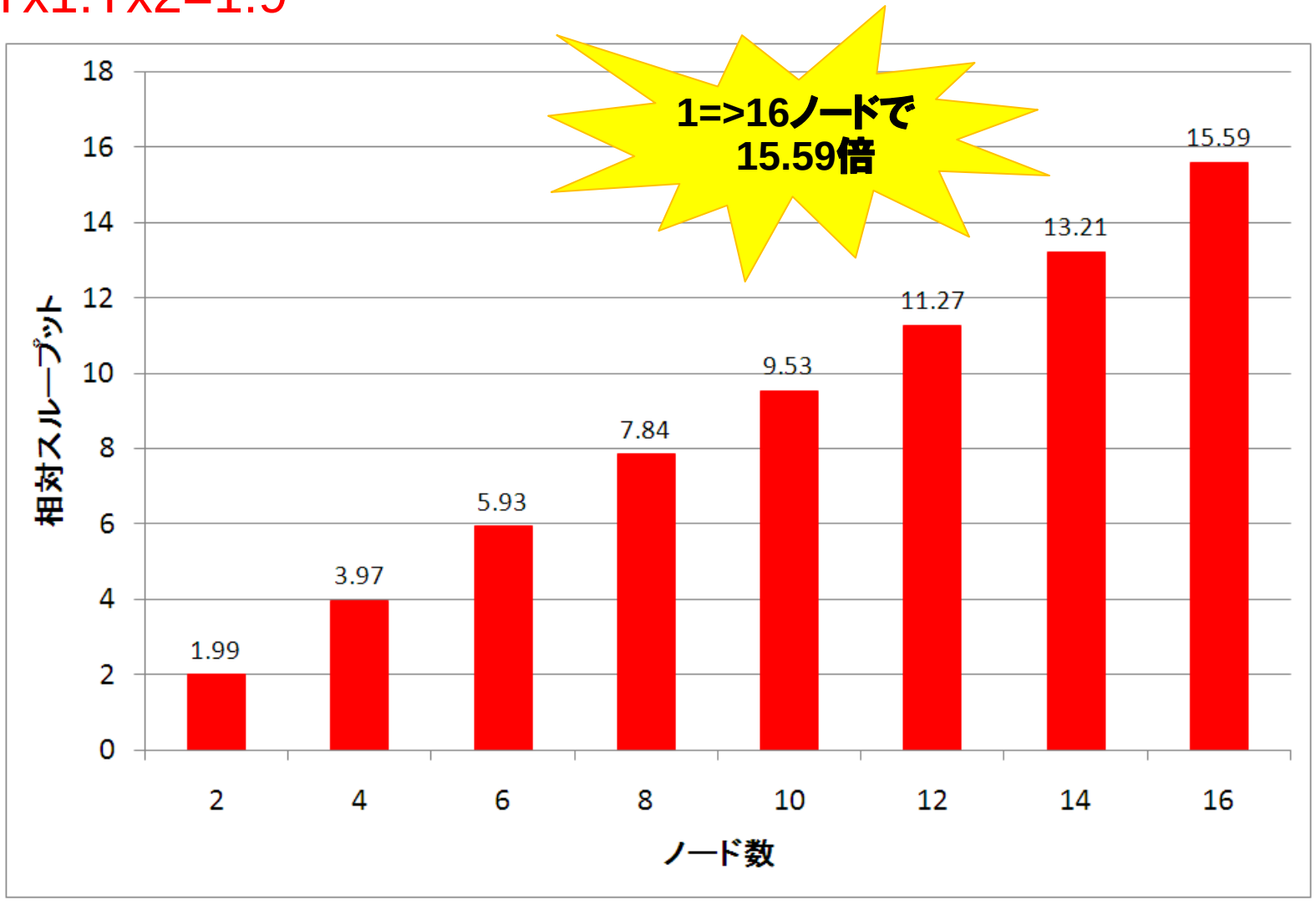
スケール・アウトの例
チューニング

使用したチューニング・ポイントの例

- キャッシュ・ヒット率を改善
- ブロックへのアクセスを分散させる
  - シーケンスのキャッシュ
  - パーティショニング
  - 逆キー索引
  - ASSM
- 余分な処理を起こさない
  - バインド変数を使用
  - 読み取り専用のテーブルをRead-Only
  - Read Mostly Lock
- SQLを並列化して性能向上

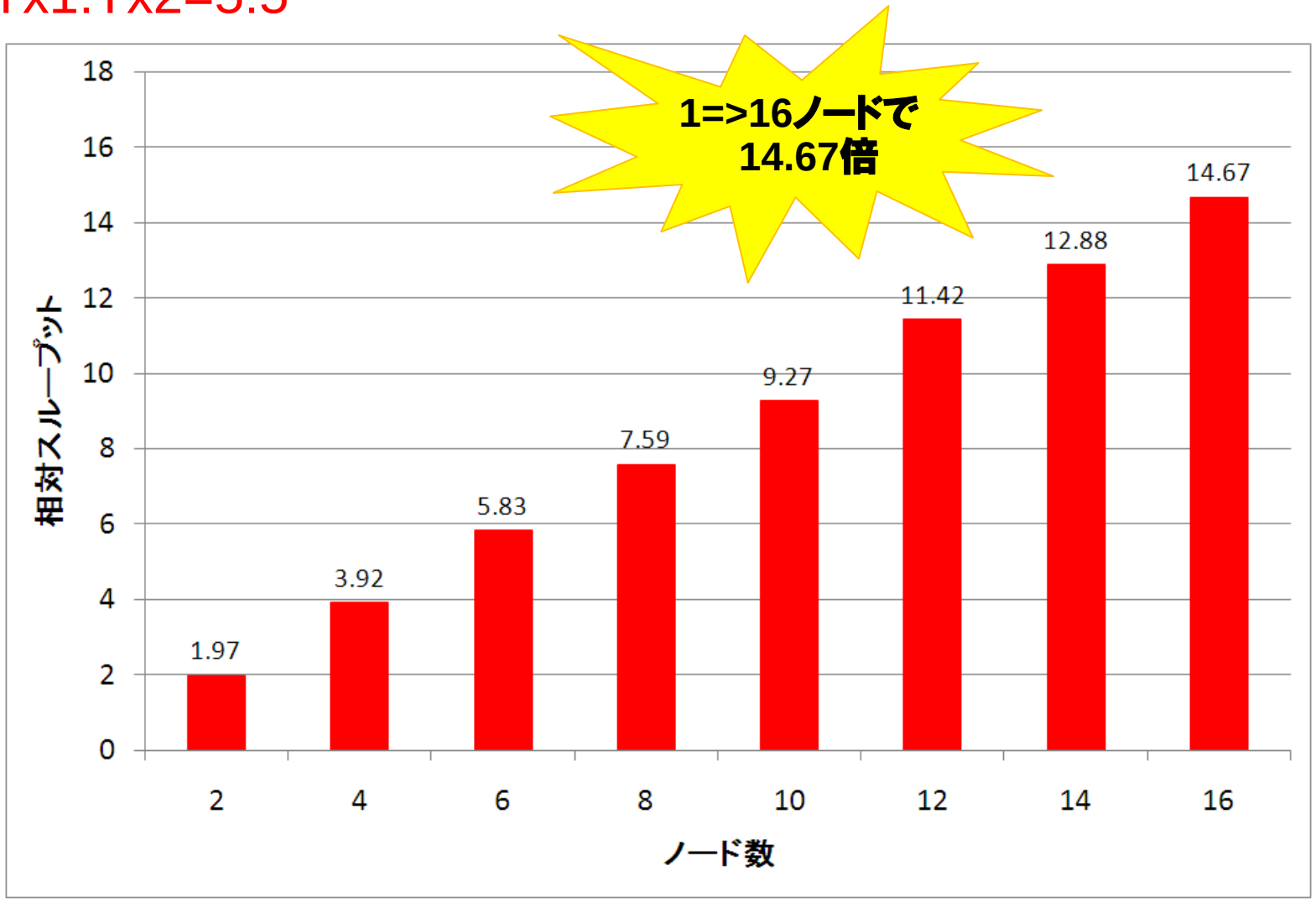
スケール・アウトの例

Tx1:Tx2=1:9



スケール・アウトの例

Tx1:Tx2=5:5



Agenda

1. はじめに
2. RAC における考慮ポイントとチューニング例
3. スケール・アウトの例
4. **まとめ**

まとめ

本セッションを終えてRACに対して持っていただけたイメージ

Cache Fusion が起こって

グローバル・キャッシュ待機イベント が発生しても

チューニングは

シングル・インスタンスと変わらない ので

同じ手法を用いれば スケールする

OTNセミナーオンデマンド

コンテンツに対する
ご意見・ご感想を是非お寄せください。

OTNオンデマンド　感想

http://blogs.oracle.com/oracle4engineer/entry/otn_ondemand_questionnaire

上記に簡単なアンケート入力フォームをご用意しております。

セミナー講師/資料作成者にフィードバックし、
コンテンツのより一層の改善に役立てさせていただきます。

是非ご協力をよろしくお願いいたします。

OTNセミナーオンデマンド

日本オラクルのエンジニアが作成したセミナー資料・動画ダウンロードサイト

ORACLE TECHNOLOGY NETWORK
OTNセミナー オンデマンド

掲載コンテンツカテゴリ(一部抜粋)
Database 基礎
Database 現場テクニック
Database スペシャリストが語る
Java
WebLogic Server/アプリケーション・グリッド
EPM/BI 技術情報
サーバー
ストレージ

100以上のコンテンツをログイン不要でダウンロードし放題

データベースからハードウェアまで充実のラインナップ

毎月、旬なトピックの新作コンテンツが続々登場

例えばこんな使い方

- 製品概要を効率的につかむ
- 基礎を体系的に学ぶ/学ばせる
- 時間や場所を選ばず（オンデマンド）に受講
- スマートフォンで通勤中にも受講可能

毎月チェック！

コンテンツ一覧 はこちら
http://www.oracle.com/technetwork/jp/ondemand/index.html

新作＆おすすめコンテンツ情報 はこちら
http://oracletech.jp/seminar/recommended/000073.html

OTNオンデマンド

ORACLE

オラクルエンジニア通信

オラクル製品に関わるエンジニアの方のための技術情報サイト

技術資料 — インストールガイド・設定チュートリアルetc. 欲しい資料への最短ルート

アクセスランキング — 他のエンジニアは何を見ているのか？人気資料のランキングは毎月更新

特集テーマ Pick UP — 性能管理やチューニングなど月間テーマを掘り下げて詳細にご説明

技術コラム — SQLスクリプト、索引メンテナンスetc. 当たり前の運用/機能が見違える!?

http://blogs.oracle.com/oracle4engineer/

オラクルエンジニア通信

oracletech.jp

ITエンジニアの皆様に向けて旬な情報を楽しくお届け

製品/技術情報
Oracle Databaseっていくら？オプション機能も見積れる簡単ツールが大活躍

セミナー
基礎から最新技術までお勧めセミナーで自分にあった学習方法が見つかる

スキルアップ
ORACLE MASTER！試験頻出分野の模擬問題と解説を好評連載中

Viva! Developer
全国で活躍しているエンジニアにスポットライト。きらりと輝くスキルと視点を盗もう

http://oracletech.jp/

oracletech

あなたにいちばん近いオラクル

Oracle Direct

まずはお問合せください

Oracle Direct

システムの検討・構築から運用まで、ITプロジェクト全般の相談窓口としてご支援いたします。
ステム構成やライセンス/購入方法などお気軽にお問い合わせ下さい。

Web問い合わせフォーム

専用お問い合わせフォームにてご相談内容を承ります。
http://www.oracle.co.jp/inq_pl/INQUIRY/quest?rid=28

※フォームの入力にはログインが必要となります。
※こちらから詳細確認のお電話を差し上げる場合がありますので
ご登録の連絡先が最新のものになっているかご確認下さい。

フリーダイヤル

0120－155－096

※月曜～金曜
9:00～12:00、13:00～18:00
（祝日および年末年始除く）

ORACLE

Hardware and Software
Engineered to Work Together


ORACLE®