ーミニサーベイー
データストリーム処理システムに関する研究動向

筑波大学システム情報工学研究科
渡辺陽介

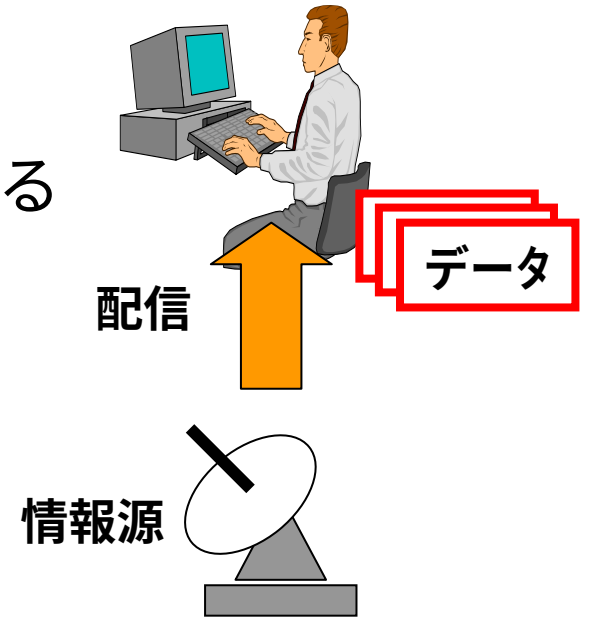
2004年03月06日　　DEWS2004 in 伊勢志摩

発表手順

1. 背景
   - ストリームとは何？
2. Continuous Query
   - 処理方式・問合せ記述
   - ウインドウ
3. 問合せ最適化
   - 適応型最適化
   - 複数問合せ最適化
4. まとめと将来の展望

データストリーム

- 時間の経過と共に変化する情報を提供する情報源
  - 金融情報，ログ，センサーデータ，オンラインニュース
- 登場の背景
  - ネットワークの発達
  - デバイスの小型化・低価格化
- データストリームの特徴
  - 情報源から能動的にデータが届けられる
  - 長期間にわたりサービスが提供される
  - 即時性が求められる

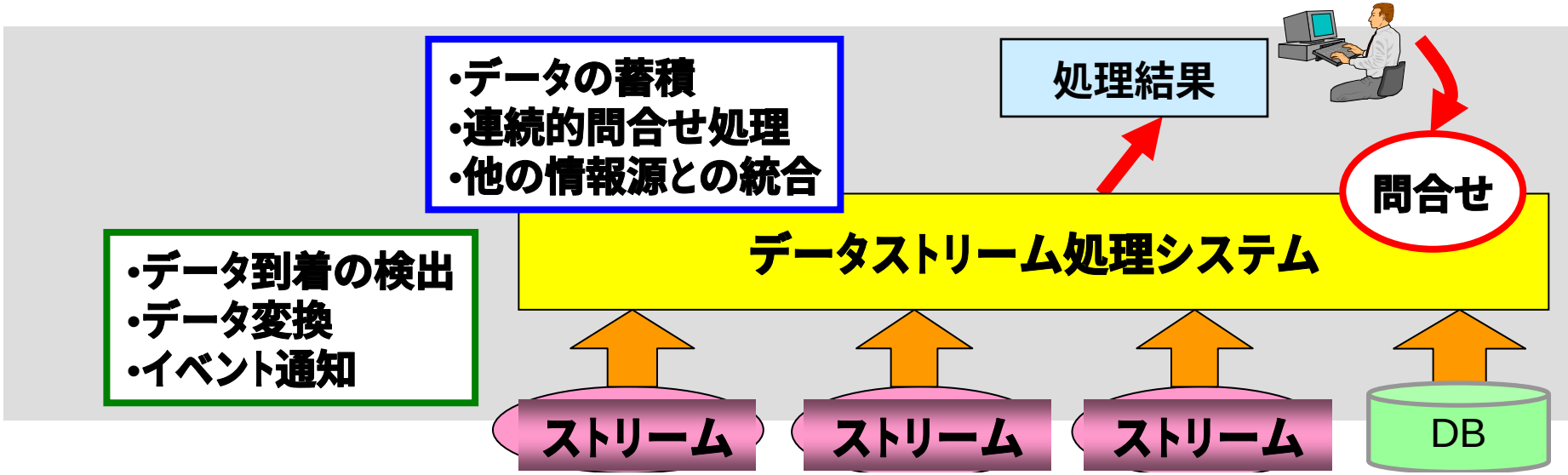


アプリケーション

用途	データ	要求
モバイル	GPSデータ	位置に応じた情報提供
ネットワーク 計算機管理	トラフィックデータ ログデータ	パケットあふれの検出 侵入検出
コンテンツ配信	HTML, XMLテキスト ニュース	フィルタリング フォーマット変換
ユビキタス	RFIDタグリーダ からの入力	物の移動・状態の検知
金融	株価情報・為替情報	リアルタイム分析
センサーネットワーク	センサーデータ （光，熱，音，映像）	監視 異常検出

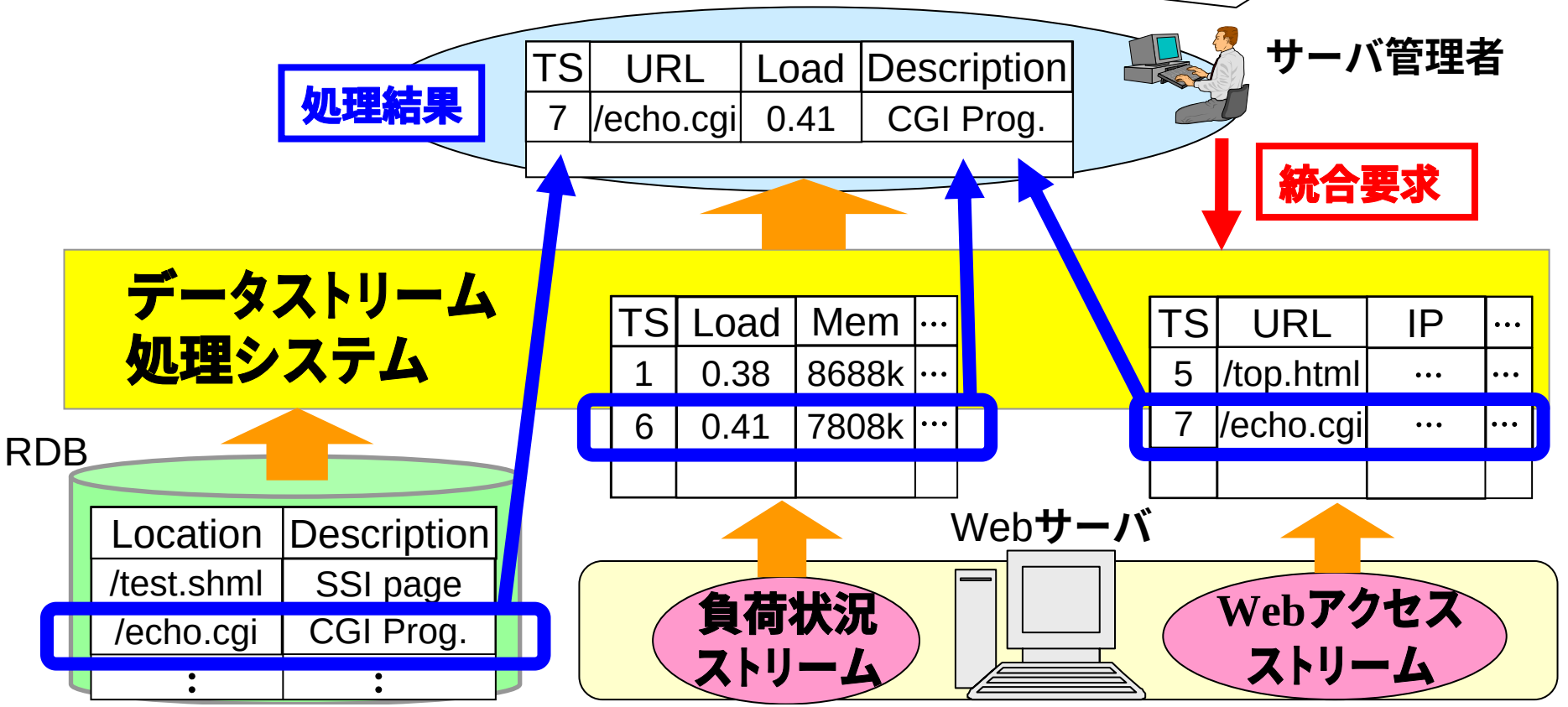
データストリーム処理システム

- データストリームの高度利用の要求の増加
  - データストリームに対する問合せ処理
    - 複数のストリームの統合，DBとの統合
    - コンテンツのフィルタリング
    - 要求に合せた通知タイミングの指定
  - データストリームからの知識抽出



利用例：Webサーバのモニタリング

登録しておいたURLへのアクセスがあったとき，
サーバの負荷が一定値以上であれば通知してほしい



これまでのＤＢＭＳとの違い

- DBMS
  - ディスクに蓄積されたデータへのアクセス
  - 与えられた問合せを一回実行して確実に結果を返す
  - 静的なコスト見積りによる問合せ最適化
- データストリーム処理システム
  - 時系列データに対するリアルタイム処理
  - 到着したデータに対し，繰り返し問合せを実行
  - アプリケーションによっては完全な問合せ結果を必要としない
  - 到着データの傾向が時間とともに変化するため，適応的な問合せ最適化が必要

主要なデータストリーム処理システム

- **STREAM**: Stanford
  - *“Continuous Queries over Data Streams”, Proc. SIGMOD2000.*
  - *“Query Processing, Resource Management, and Approximation in a Data Stream Management System”, Proc. CIDR 2003.*
- **TelegraphCQ**: Berkely
  - *“TelegraphCQ: Continuous Dataflow Processing for an Uncertain World”, Proc. CIDR 2003.*
  - *“PSoup: a system for streaming queries over streaming data”, VLDB Journal 2003.*
- **Aurora**: Brown, MIT
  - *“Aurora: a new model and architecture for data stream management”, VLDB Journal 2003.*

主要なデータストリーム処理システム

- **NiagaraCQ**: Wisconsin
  - *“NiagaraCQ: A Scalable Continuous Query System for Internet Databases”, SIGMOD 2000.*
  - *“Design and Evaluation of Alternative Selection Placement Strategies in Optimizing Continuous Queries”, ICDE 2002.*
- **Gigascope**: AT&T
  - *“Gigascope: A Stream Databases for Network Applications”, Proc. SIGMOD 2003.*
- **PeerCQ**: Georgia Tech.
  - *“PeerCQ: A Decentralized and Self-Configuring Peer-to-Peer Information Monitoring System”, ICDCS 2003.*
- などなど

発表手順

1. 背景
   - ストリームとは何？
2. Continuous Query
   - 処理方式・問合せ記述
   - ウインドウ
3. 問合せ最適化
   - 適応型最適化
   - 複数問合せ最適化
4. まとめと将来の展望

Continuous Query

- データストリームからの到着データへ繰り返し問合せを適用する処理方式
  - 到着データをタイムスタンプが付加された無限のタプル列とみなす
  - ※今回はXMLストリームの話はしない
- 問合せ記述形式
  - SQLライクな構文による記述[CQL, GSQLなど]
  - GUIによる問合せ指定[Aurora]
- 問合せの実行タイミングによる分類
  - 到着型(Change-based)： 新規情報の到着時に実行
  - タイマー型(Timer-based)： タイマーにより定期的に実行

CQL: Continuous Query Language

- StanfordのSTREAMの問合せ言語
  - *“The CQL Continuous Query Language: Semantic Foundations and Query Execution”, TR.*
- 特徴
  - SQLライクな構文を持つ問合せ言語
    - リレーションとストリームを扱うことが可能
  - スライディングウインドウの宣言（後述）
    - 3種のウインドウ(Time-based,Tuple-based, Partitioned)

スライディングウインドウ

- ストリームから有限のリレーションを抽出
  - ストリームからはデータが無限に到着する
  - 過去全てのデータを処理対象にはしない
- **Time-based Window**
  - タプルを保持する期間を指定
  - 「過去○○時間分のタプルに対して～して欲しい」
- **Tuple-based Window**
  - ウインドウに保持できるタプルの上限を指定
  - 「最新の○○件のデータに対して～して欲しい」
- **Partitioned Window**
  - 特定の属性の値でグループ化してタプルを保持
  - 「属性○○の各値ごとに最新の×個のタプルが欲しい」

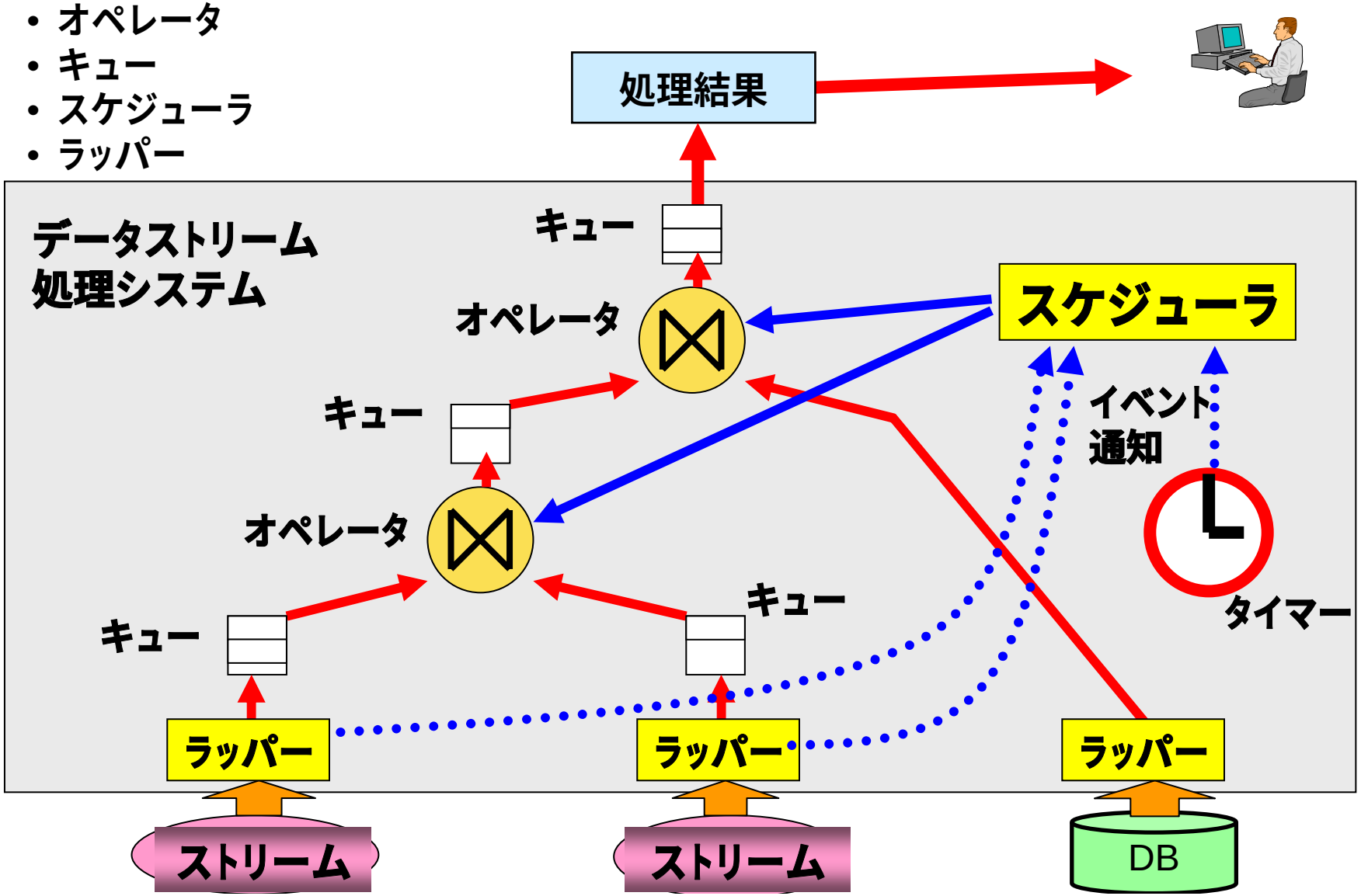
CQL問合せ例

- Webサーバのモニタリング例について記述
  - Webアクセス：Access(TS, URL, IP)
  - サーバ負荷：Status(TS, Load, Mem)
  - URLリスト：URLs(Location, Description)
  - 問合せ記述

```
Select  Access.URL, Status.Load, URLs.Description
From Access[NOW], Status[Range 5Minute], URLs
Where  Access.URL=URLs.Location
        AND Status.Load >= 0.40
```

標準的なシステムアーキテクチャ

- オペレータ
- キュー
- スケジューラ
- ラッパー



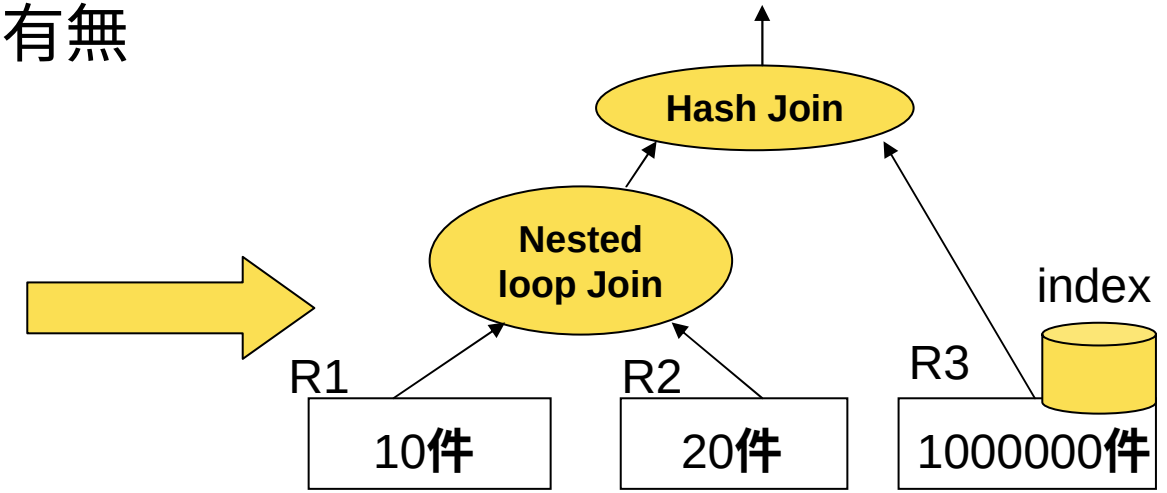
発表手順

1. 背景
   - ストリームとは何？
2. Continuous Query
   - 処理方式・問合せ記述
   - ウインドウ
3. 問合せ最適化
   - 適応型最適化
   - 複数問合せ最適化
4. まとめと将来の展望

DBMSにおける問合せ最適化

- 問合せ記述から実行効率のよいプランを生成する
  - 実行プランの候補の中からコスト最小なものを選出
  - ディスクIOの回数を減らすことが目的
- コスト見積りの基準
  - テーブルや中間結果のサイズ
  - 演算の選択率
  - インデックスの有無

```
Select  *
From  R1, R2, R3
Where R1.a1 = R2.a1
   AND R2.a2=R3.a2
```

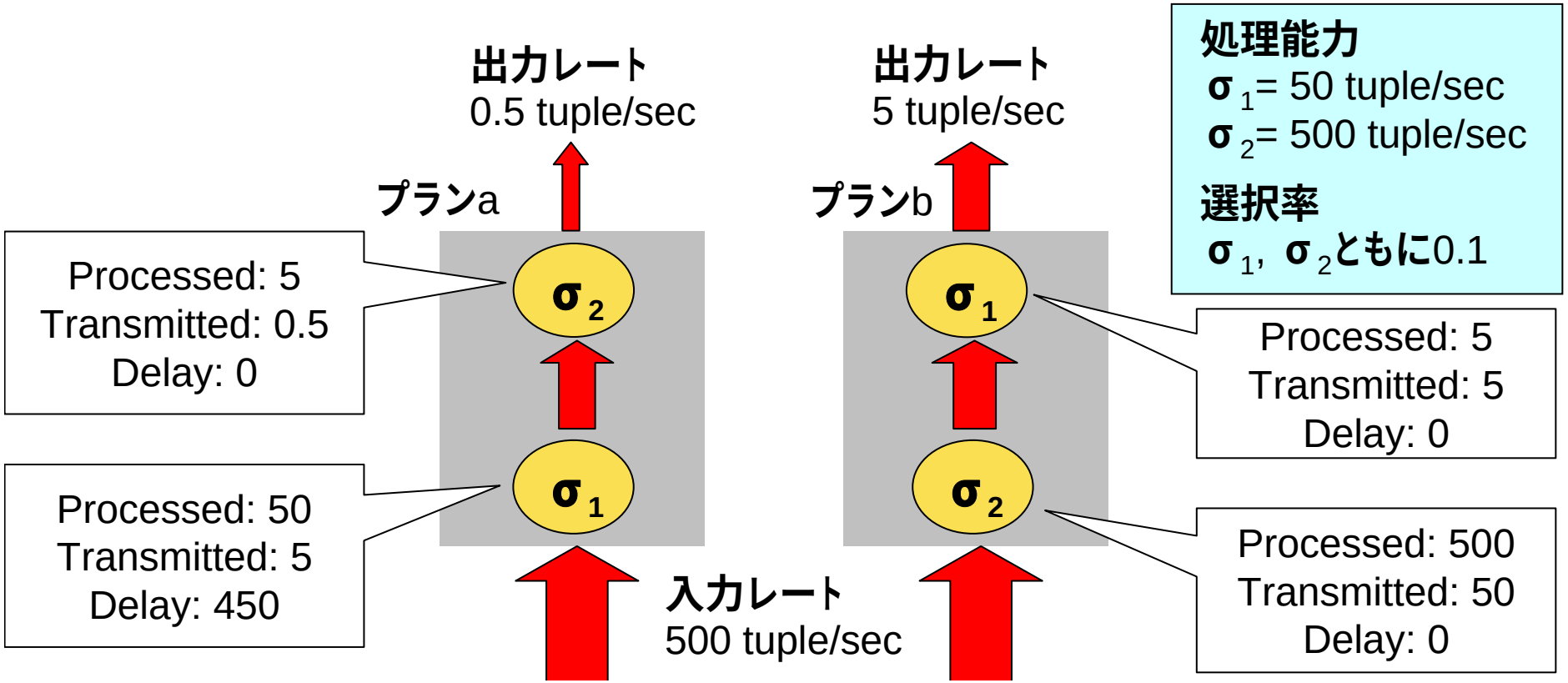


ストリームにおける問合せ最適化

- モチベーション
  - DBMSの問合せ最適化とは前提が大きく異なる
    - 何をすることが最適化か？ 何を基にコスト計算をするのか？
  - ストリームのデータは時間と共に特性や傾向が変化
- 静的な最適化： レートに基づいた方式
  - ストリームの特性が長期間固定な場合
  - 固定の最適なプランをコスト計算により求める
  - 定期的にコスト再計算を行い，最適化をやりなおす
- 動的な最適化： Eddyを用いた方式
  - ストリームの特性が頻繁に変化する場合
  - 固定のプランを持たない！

レートに基づいた問合せ最適化

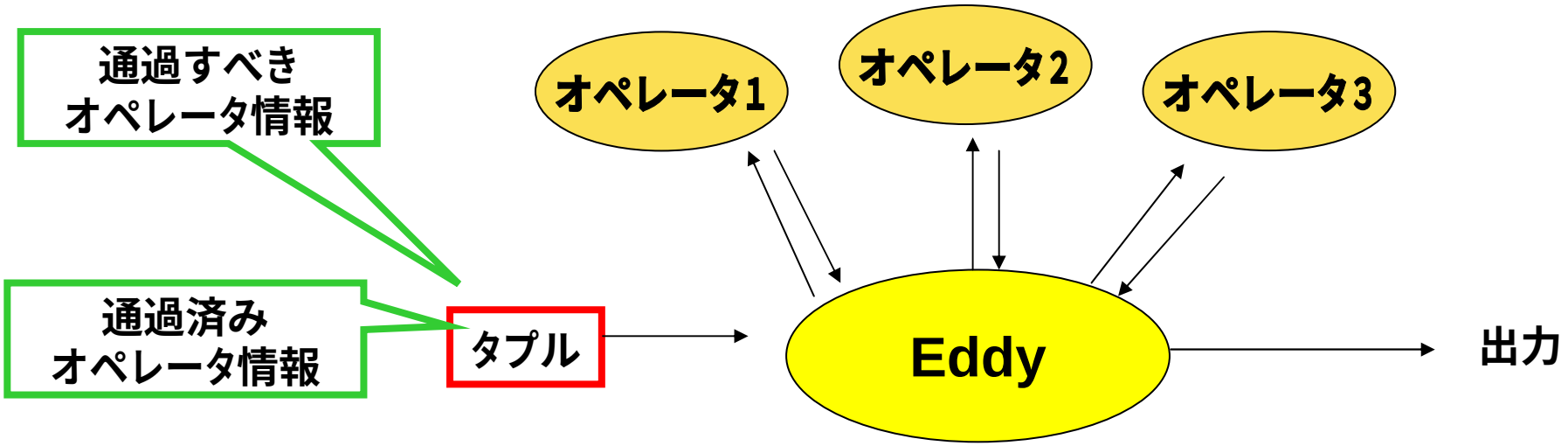
- *“Rate-Based Query Optimization for Streaming Information Sources”, Viglas ら SIGMOD 2002.*
  - タプルの出力レートを最大化するプランを選択
  - ストリームの到着レートに基づいてコスト計算を行う



Eddyを用いた適応型処理

- Eddy
  - タプルをオペレータへルーティングするモジュール
  - 固定のプランを持たず，タプル単位で処理経路を選択
  - *“Eddies: Continuously Adaptive Query Processing”, SIGMOD 2000.*
- タプルを次にどのオペレータに渡すかは以下により決定
  - 各オペレータのキューの空き状況
  - 各オペレータの選択率

→単位時間に処理するタプル数（スループット）の向上

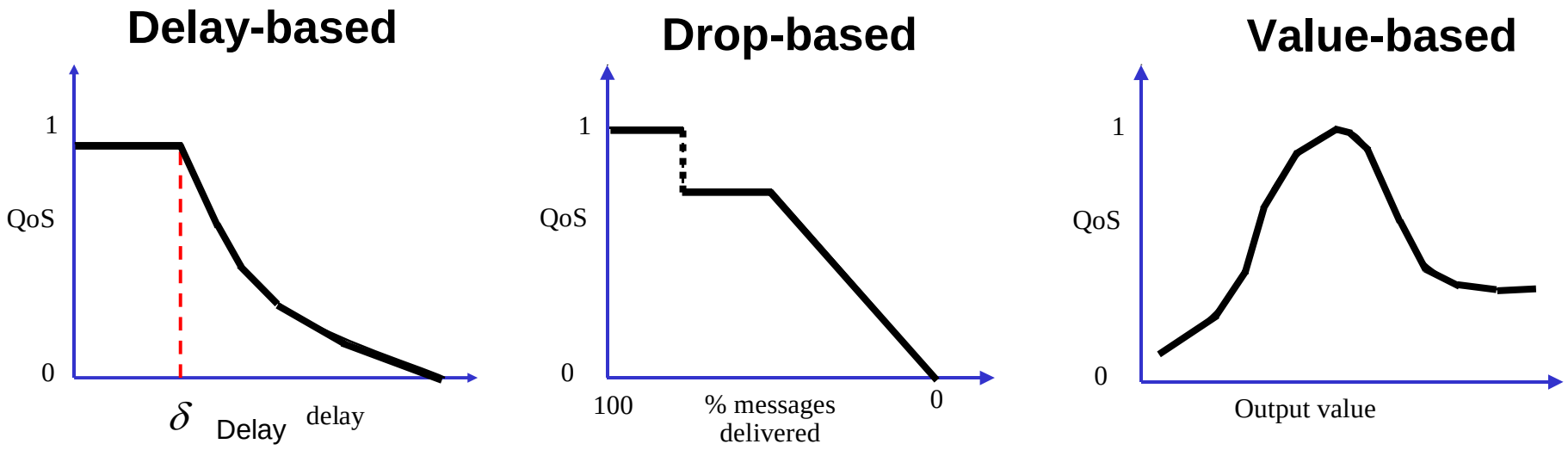


Load Shedding

- 背景
  - ストリームデータの到着レートの変化により，システムの負荷が一時的に過剰になる
- Load Shedding:
  - システムがオーバーロードを検出したら，入力タプルを落とし，負荷を軽減する
  - アプリケーションによっては，必ずしも全データが必要でないこともあるため
- 提案されている手法
  - Random tuple drops [STREAM, Aurora]
    - ランダムに入力を削減
  - Semantic load shedding [Aurora]
    - QoS指定に合わせてフィルタ演算を挿入

AuroraにおけるQoS指定

■問合せ結果をどの程度有益とみなすか



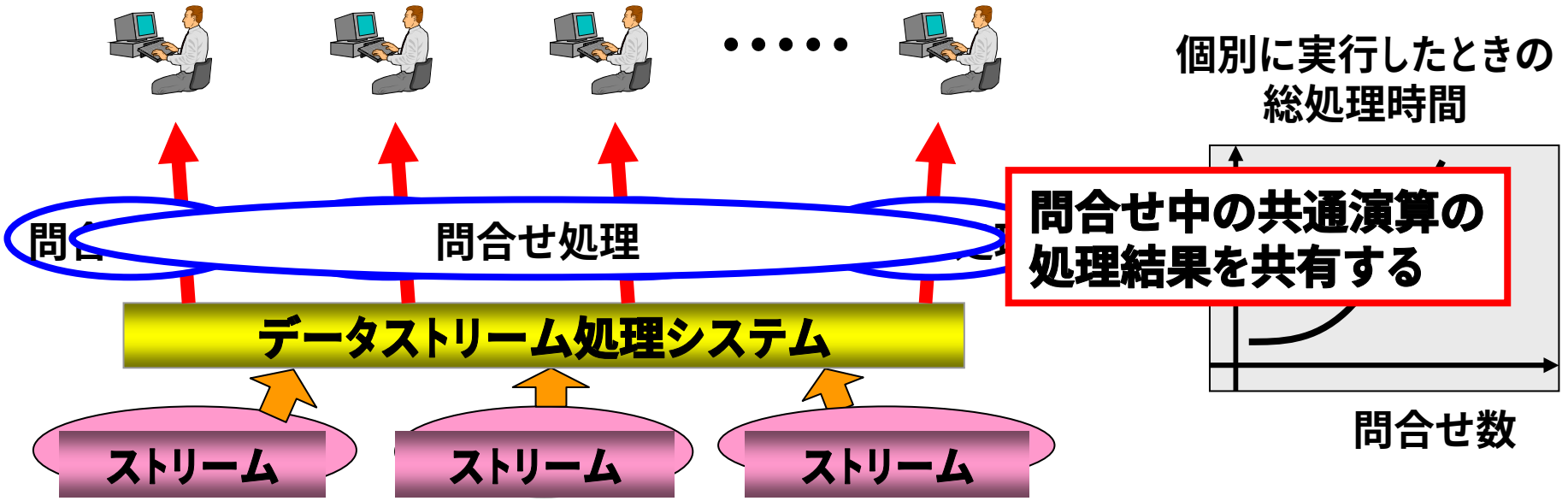
- Delay-based： スケジューリング優先度
  - 遅延がδ未満なら有益．それ以降は経過時間に伴い減少
- Drop-based： Random tuple drops用
  - 結果が100%配信されれば有益，結果が減るほど低い
- Value-based： Semantic load shedding用
  - いくつかの値にだけ着目したいときなど
  - 横軸上の重要な値に高いQoSの値を指定する

発表手順

1. 背景
   - ストリームとは何？
2. Continuous Query
   - 処理方式・問合せ記述
   - ウインドウ
3. 問合せ最適化
   - 適応型最適化
   - 複数問合せ最適化
4. まとめと将来の展望

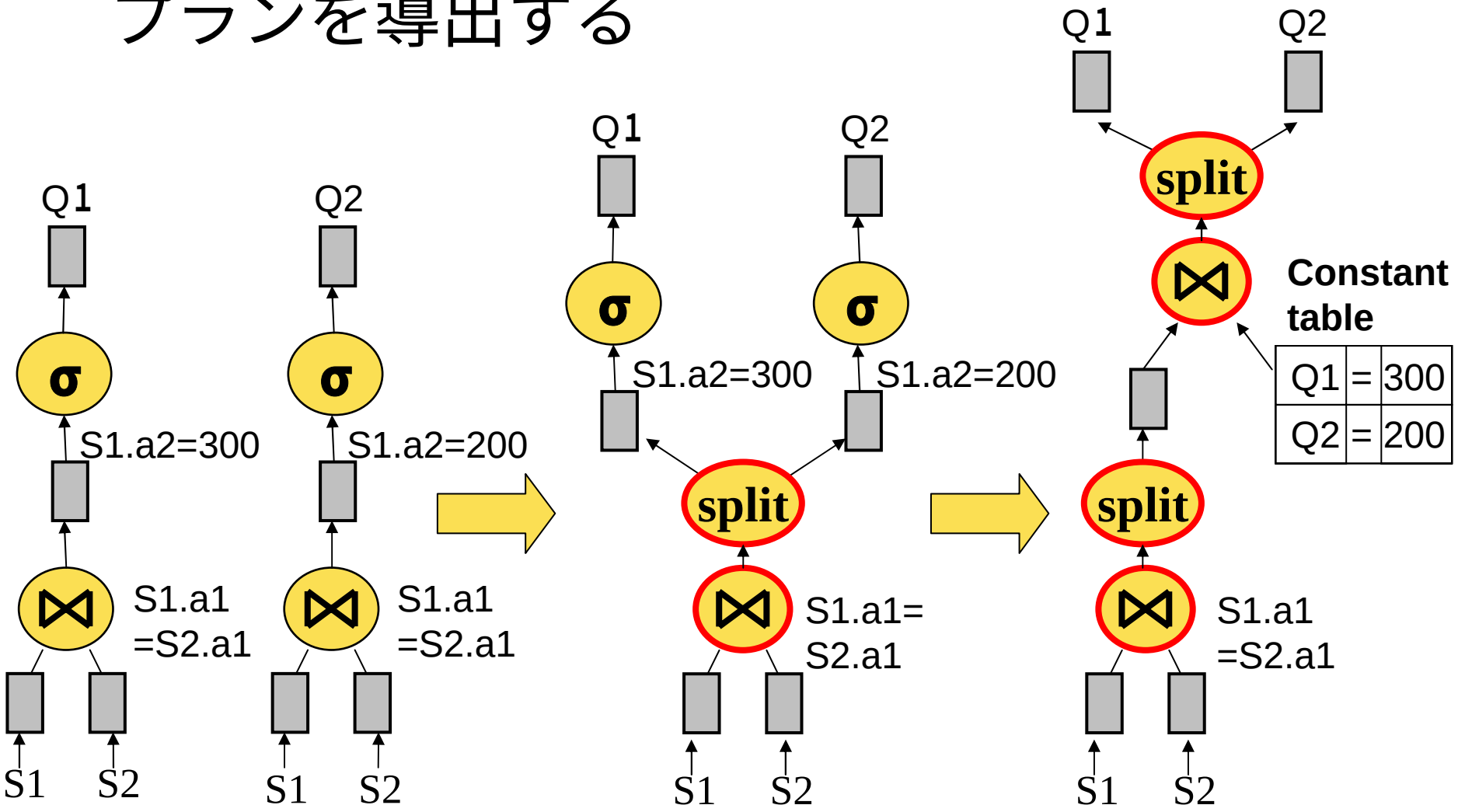
複数問合せ最適化 (Multiple Query Optimization)

- 利用者が増加し，大量の問合せ処理要求が発生
  - 問合せ処理にかかる時間が増大
  - 大量の問合せを効率的に処理可能な方式の必要性
- Continuous queryに対する複数問合せ最適化方式



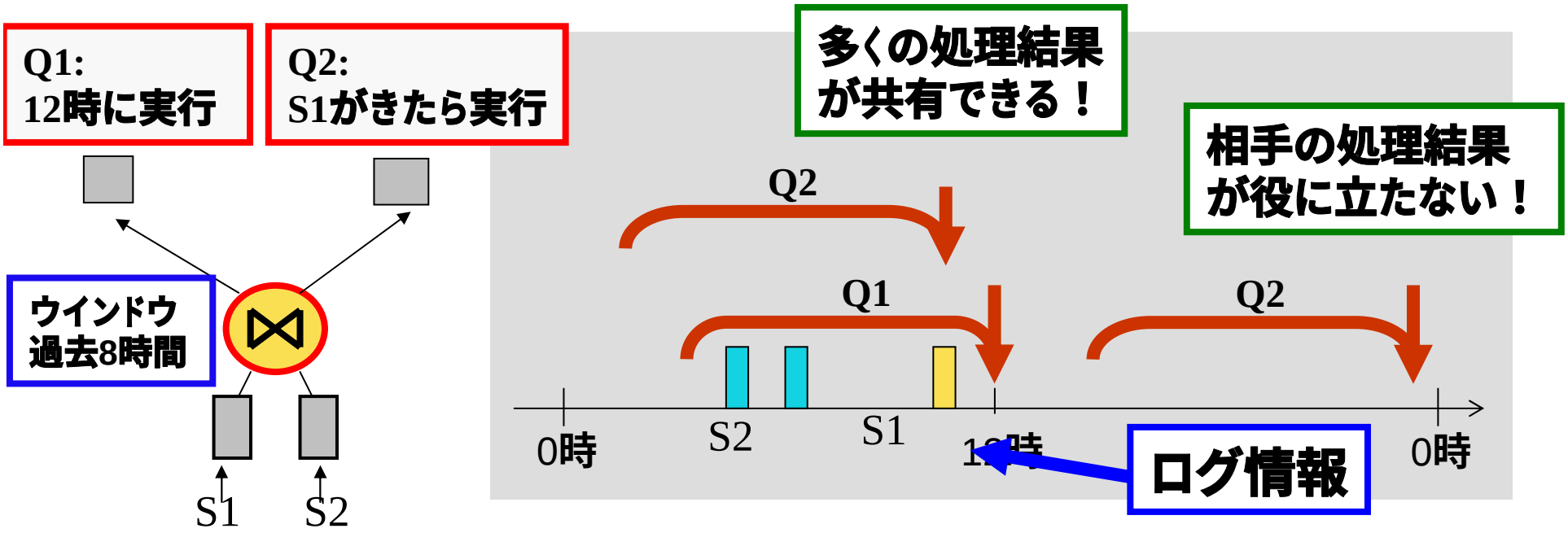
NiagaraCQにおける複数問合せ最適化

- 問合せ中の共通演算の処理結果を共有するプランを導出する



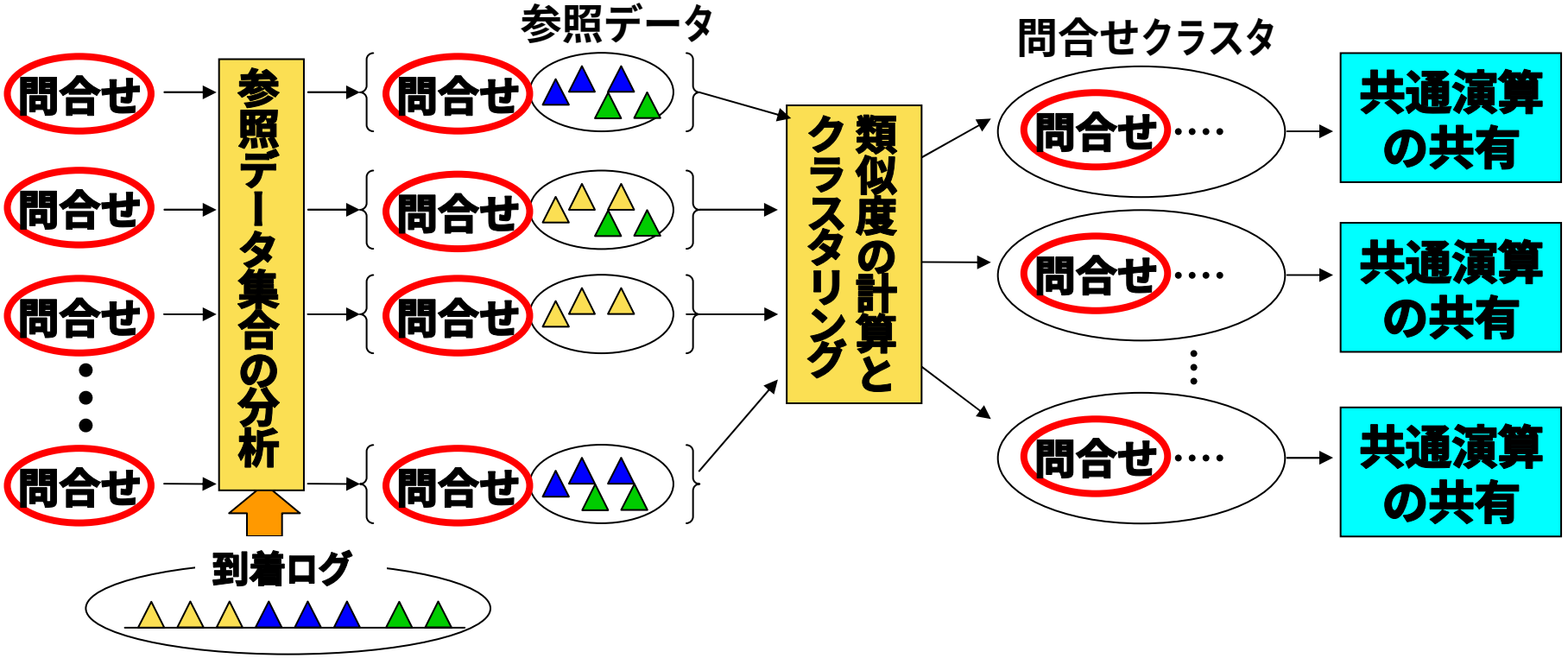
複数問合せ最適化[渡辺,北川 DEWS2003]

- 実行タイミングが問合せごとに異なる場合
  - 共通演算でも異なる範囲のデータを参照する可能性がある
  - 共有できるデータが本当に生成されるか調べなければならない
- 到着型問合せはいつ実行されるのか不明
  - ログ情報による実行のシミュレーションが必要



複数問合せ最適化[渡辺,北川 DEWS2003]

- 到着ログを元に問合せの実行タイミングと参照データを分析
- 参照データ集合をもとに問合せを分類
  - 共通の参照データの割合が高いほど問合せ間で共有可能な結果を多く含む



まとめ　～データストリーム処理システム～

システム名	問合せのタイプ	問合せプラン導出	複数問合せ最適化	Load Shedding
STREAM	到着型 タイマー型	固定のプラン	○ 確実に同時に実行される演算	Random
TelegraphCQ	到着型	eddyによる動的最適化	○ 確実に同時に実行される演算	×
Aurora	到着型	固定のプラン 定期的な最適化	○ 確実に同時に実行される演算	Random Semantic
NiagaraCQ	到着型 タイマー型	固定のプラン	△ 実行タイミングを考慮せず	×
渡辺,北川	到着型 タイマー型	固定のプラン 定期的な最適化	◎ 実行タイミングと参照範囲を考慮した問合せのグループ化	×

将来の展望

- DBMS，DataWarehouse技術との連携
  - DBMS
    - 静的な業務データ
  - DataWarehouse
    - 履歴データ
  - データストリーム処理システム
    - リアルタイムデータ
- アプリケーション開発環境の整備
- 性能評価のためのベンチマーク
  - Linear road benchmark

    http://www-db.stanford.edu/stream/cql-benchmark.html