【VLDB 2010勉強会】

Session 6: Database Design

担当：欅惇志（NAIST）

Session 6: Database Design

- CRIUS: User-Friendly Database Design
  - Li Qian (University of Michigan), Kristen LeFevre (University of Michigan), H. Jagadish (University of Michigan)
  - Span table の提案と schema evolution 補助のインタフェースの開発
- CoPhy: A Scalable, Portable, and Interactive Index Advisor for Large Workloads
  - Debabrata Dash (ArcSight), Neoklis Polyzotis (UC Santa Cruz), Anastasia Ailamaki (Ecole Polytechnique Fédérale de Lausanne (EPFL)
  - BIP を利用して index の configuration を高速に発見
- Compression Aware Physical Database Design
  - Hideaki Kimura (Brown University), Vivek Narasayya (Microsoft Research), Manoj Syamala (Microsoft Research)
  - Index の圧縮を考慮した configuration の決定
- Structure-Aware Sampling: Flexible and Accurate Summarization
  - *Edith Cohen (AT&T Labs), Graham Cormode (AT&T Labs), Nick Duffield (AT&T Labs)*
  - データ構造を推測した上でサンプリングを行う手法の提案

※ 図・表は論文もしくはプレゼン資料から引用

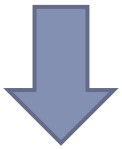
CRIUS: User-Friendly Database Design

- 目的
  - 動的なスキーマ変更の必要性
    - 非専門家によるスキーマ設計
    - 現在のスキーマで表現できないデータの追加
- 概要
  - スキーマ変更のための代数，インタフェース
  - 関数従属性 (FD) の発見，FD を元に自動誤り訂正

簡単，低コスト

- Span table
  - Spreadsheet ライクなデータ構造
    - 単一，入れ子構造のテーブル
  - RDB で階層構造を表すには複数テーブルに分割
    - End user には難しい

Name	City	Address
Mary	Chicago	Bishop
Keith	Arbor	Plymouth



Name	City	[Address]
		Address
Mary	Chicago	Bishop
Keith	Arbor	Plymouth
		Main

CRIUS: User-Friendly Database Design

- Span table algebra
  - スキーマ変更操作
    - Import, export, float, sink
  - データ操作
    - Adding / dropping columns
    - Inserting / deleting / updating tuples

Name	City	Address
Mary	Chicago	2364 Bishop
Keith	Ann Arbor	101 Plymouth
Keith	Ann Arbor	202 Main

Sink(Address)

Name	City	[Address]
		Address
Mary	Chicago	2364 Bishop
Keith	Ann Arbor	101 Plymouth
		202 Main

Import(City)

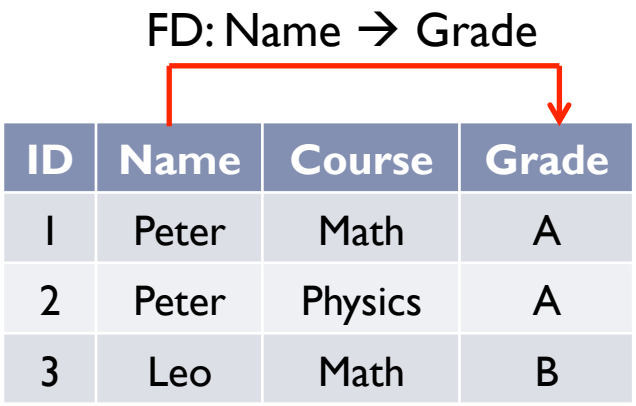
Name	[Address]	
	City	Address
Mary	Chicago	2364 Bishop
Keith	Ann Arbor	101 Plymouth
	Ann Arbor	202 Main

Export(City)

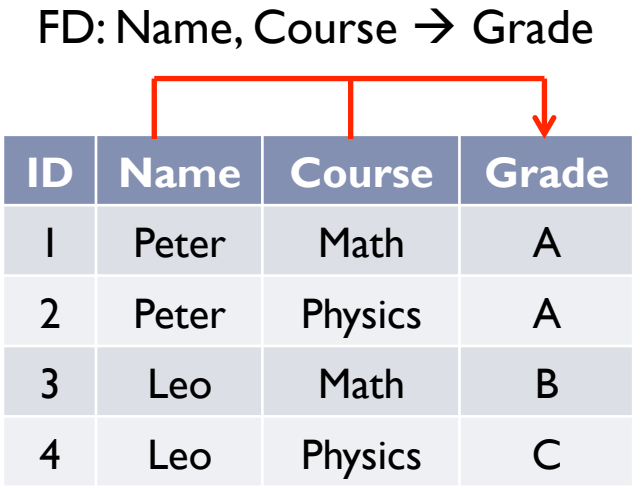
Name	City	[Address]
		Address
Mary	Chicago	2364 Bishop
Keith	Ann Arbor	101 Plymouth
		202 Main

CRIUS: User-Friendly Database Design

- FD 関連の機能
  - FD の推測
  - 入力の誤り検知 / FD 更新
  - Incremental FD induction
    - コスト削減のためのアルゴリズム
    - Name → {(1, 2), (3, 4)}
    - {Name, Course} → {(1), (2), (3), (4)}
    - 値変更による FD 更新時にも利用



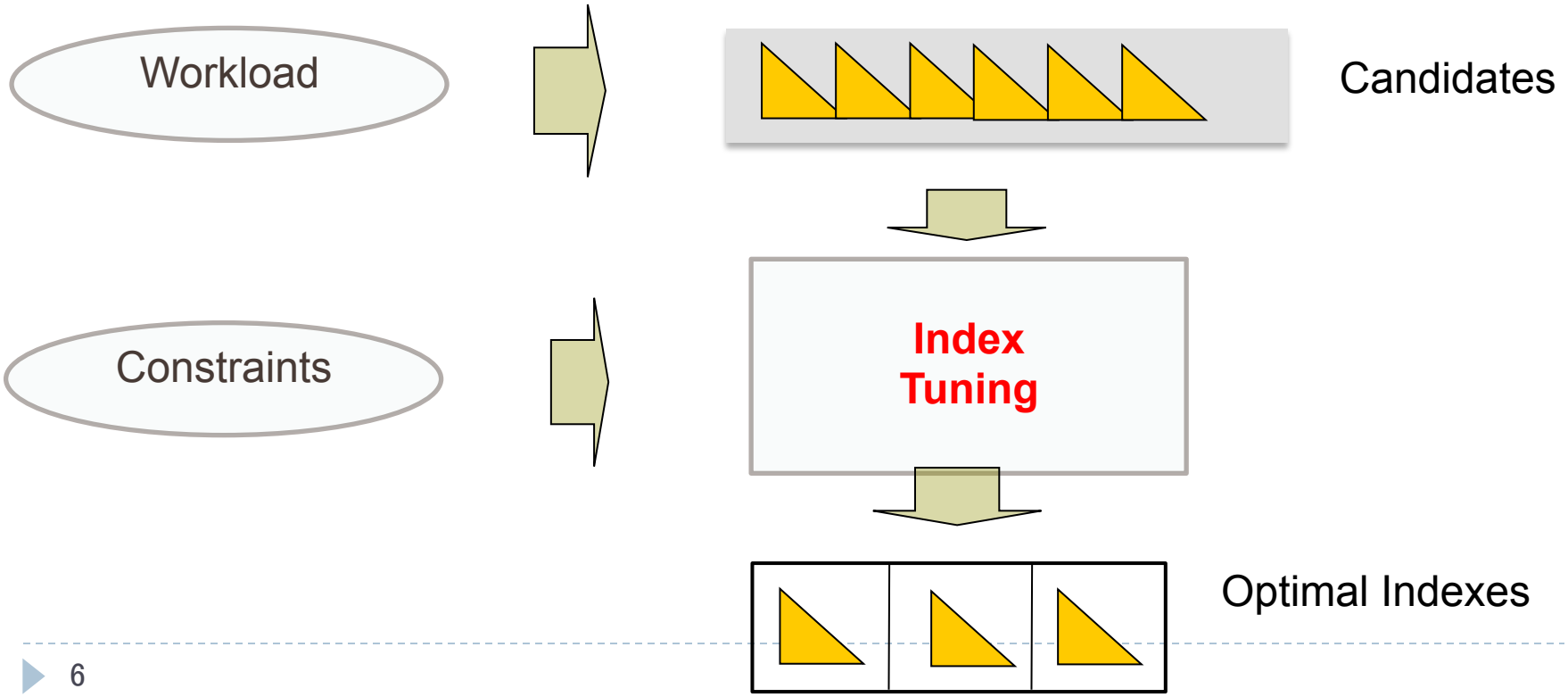
ID	Name	Course	Grade
1	Peter	Math	A
2	Peter	Physics	A
3	Leo	Math	B



ID	Name	Course	Grade
1	Peter	Math	A
2	Peter	Physics	A
3	Leo	Math	B
4	Leo	Physics	C

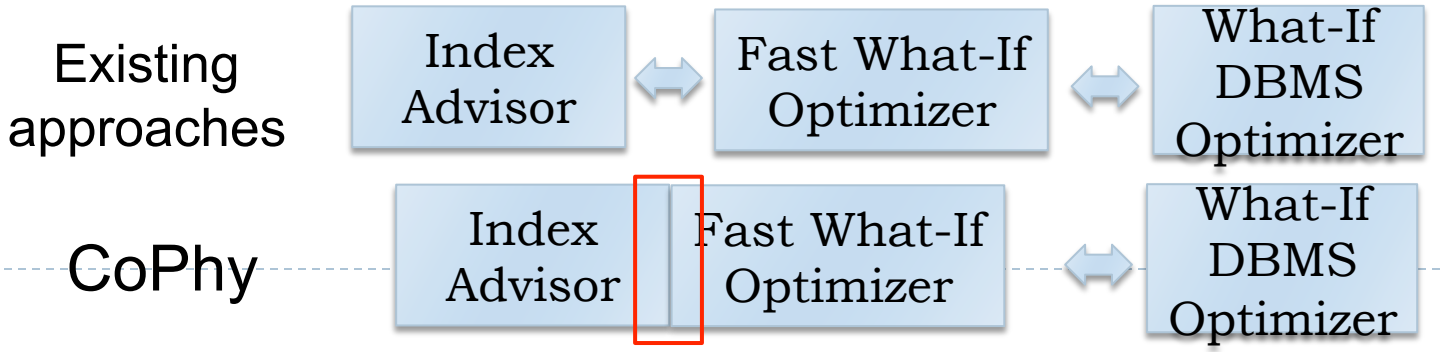
CoPhy: A Scalable, Portable, and Interactive Index Advisor for Large Workloads

- 概要
  - Workload (クエリセット) に見合った index tuning
    - DBA の作業の多くが tuning (What-if optimize)
    - Index の候補，クエリ，制約が大量 → 最適解の発見が困難



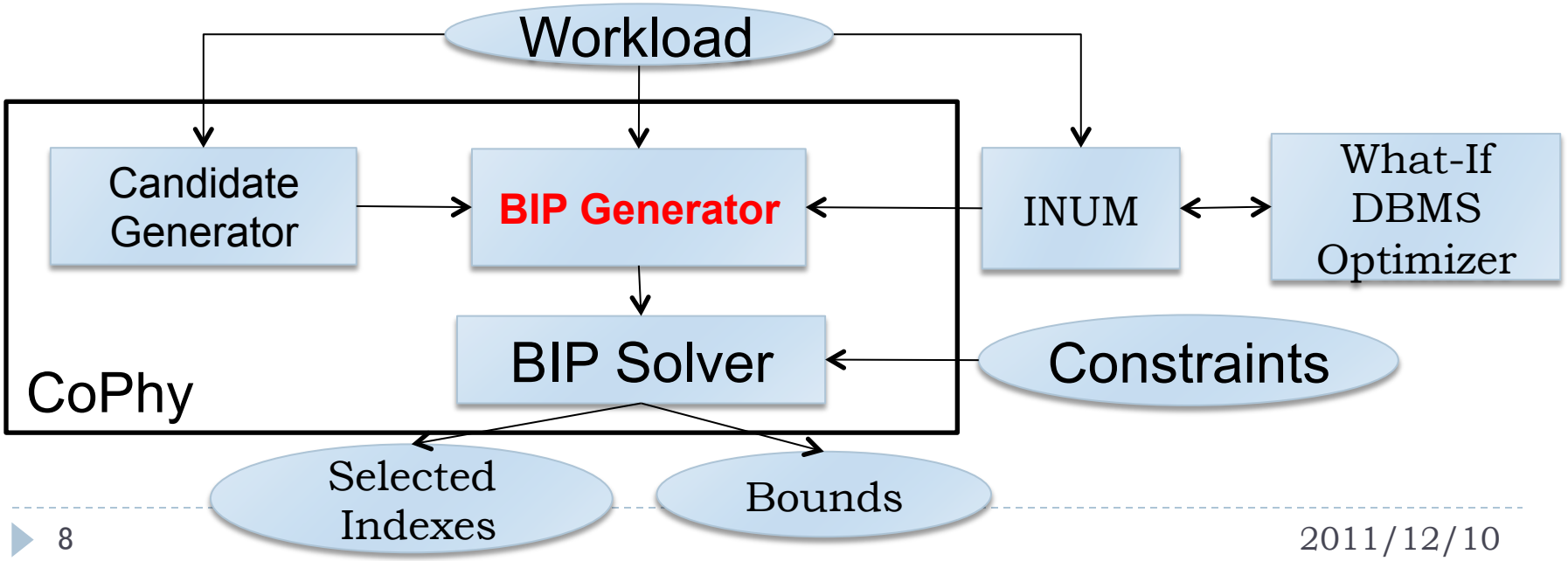
CoPhy: A Scalable, Portable, and Interactive Index Advisor for Large Workloads

- 概要
  - Workload (クエリセット) に見合った index tuning
    - DBA の作業の多くが tuning (What-if optimize)
    - Index の候補，クエリ，制約が大量 → 最適解の発見が困難
- 解決法
  - 線形計画法 (BIP) を利用 (fast what-if optimize)
    - 一つのテーブルから一つの index を作成
    - Index の候補，制約から数式を作成 (solver で解く)
    - (NP 完全問題だが) real world のデータに対しては効果的
- 既存研究との違い



CoPhy: A Scalable, Portable, and Interactive Index Advisor for Large Workloads

- インタフェースの特徴
  - DBA が満足した時点で 最適化を終了できる
  - Index の候補，クエリ，制約の追加 / 削除が容易
    - 即座に結果を取得
    - インタラクティブなやりとり



Compression Aware Physical Database Design

- 概要
  - DBMS においてデータの圧縮は一般的
    - ストレージの容量，I/O の削減
    - 圧縮，解凍を考慮せずに index 作成による問題
      - 有用な index 見逃す，CPU オーバーヘッド発生 (後述)
    - 圧縮後のデータサイズ，圧縮 / 解凍コストを考慮した最適化
- 既存研究のアプローチ
  1. Workload を用いて what-if analysis
  2. Index の作成，圧縮
  - index サイズ < budget の限り繰り返し

Budget 100MB

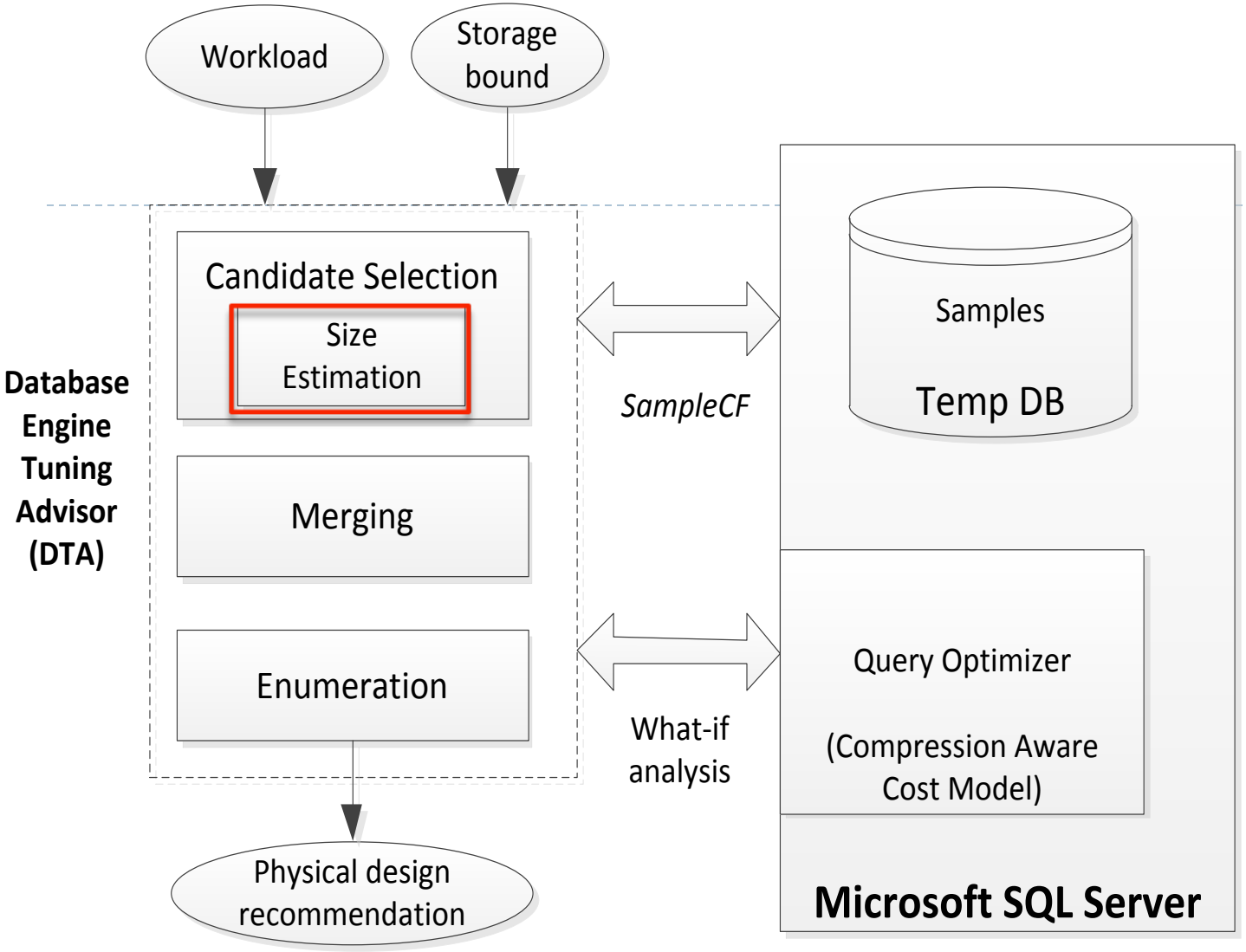
I_1: 95MB → 50MB

I_2: 170MB → 90MB 除外

Budget 250MB

I_3: 240MB → 120MB

Insert / update 時の圧縮，解凍コスト高



- サイズ推測
  - サンプリング
  - Null suppression, local 辞書による圧縮
    - Order independent / dependent
  - 推測精度とコストはトレードオフの関係
- Index 選択
  - グラフ探索アルゴリズム
  - 空間と時間はトレードオフ

Structure-Aware Sampling: Flexible and Accurate Summarization

- 概要
  - データ分析する際に元データが大きいという場合が多々
    - Summary データを用いて分析したい → サンプリング！
  - ランダムサンプリングはデータ (keys) の構造を考慮せず
    - 構造: 地理データ，ネットワークデータの階層，タイムスタンプの順序
    - 複数範囲を持つ範囲問合せでは適切にサンプリングできず
  - 評価実験：既存の分散最適化手法と比較
    - 多少の速度低下
    - 大幅に精度上昇