Estimating Join Selectivities using Bandwidth-Optimized Kernel Density Model

M. Kiefer, M. Heimel, S. Breß, V. Markl

Problem & Motivation

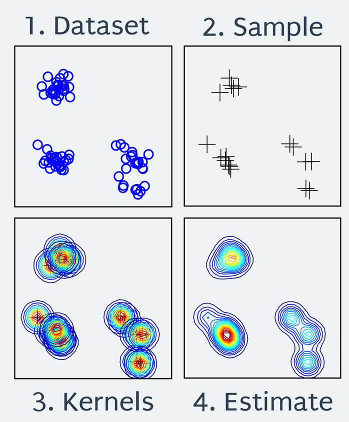
- Task: Estimating selectivities for queries of queries of the form

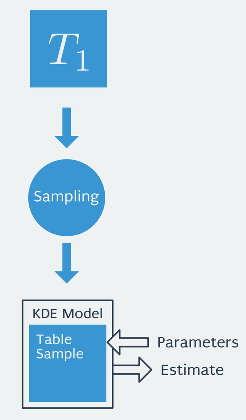
$$|\sigma_{c_1}(R_1) \bowtie_{R_1.A=R_2.A} \sigma_{c_2}(R_2)|$$

- 1D statistics + Independence Assumption (IA) are commonly used in practice
  - But IA is often violated in real-world data
  - Potentially results in suboptimal query plans
- Lifting IA requires multidimensional statistics that are
  - accurate
  - efficiently computable
  - easy to maintain and construct
- Prior approaches do not provide all these characteristics
- Generalization to n-way joins and selections on join predicate are covered in the paper

KDE for Selectivity Estimation

- Kernel Density Estimation (KDE)
  - Multivariate probability density estimation
  - Based on a uniform data sample
  - Smoothing by centering kernel functions on sample points
  - Smoothing controlled by bandwidth parameter
- KDE has been applied to range filters over base tables
  - Good accuracy
  - Hybrid between sampling and histograms
    - Bandwidth selection based on query feedback
    - Learning estimator
  - Efficiently trainable and evaluable
  - Suitable for GPU acceleration
- In this publication: Extension to joins that are subject to selections

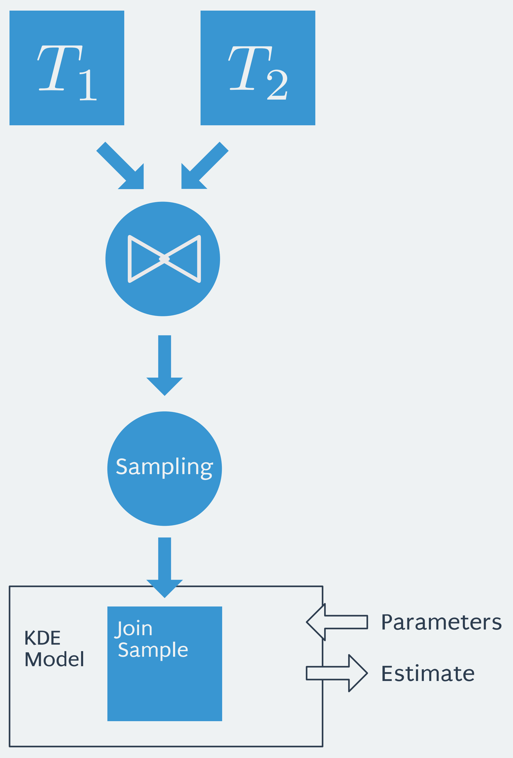




Method 1: Join Model

- KDE model constructed from the join result
  - Sampling from the join result
  - Requires exact or a good estimate of the join size
- Joins: Handled implicitly by sampling
- Selections: Handled explicitly during estimation

- (+) Very accurate estimates
- (+) Cheap model evaluation
- (−) Limited to one particular join
- (−) More expensive model construction and maintenance



Method 2: Table Model

- Estimates are computed by combining base table KDE models
  - Joining the estimated distributions
- Joins: Handled explicitly during estimation
- Selections: Handled explicitly during estimation
- Naive implementation: Pass over the cross product of samples
  - Pruning techniques are required

- (+) Cheap model construction & maintenance
- (+) General model: Supports joins, base table selections
- (−) More expensive estimate computation

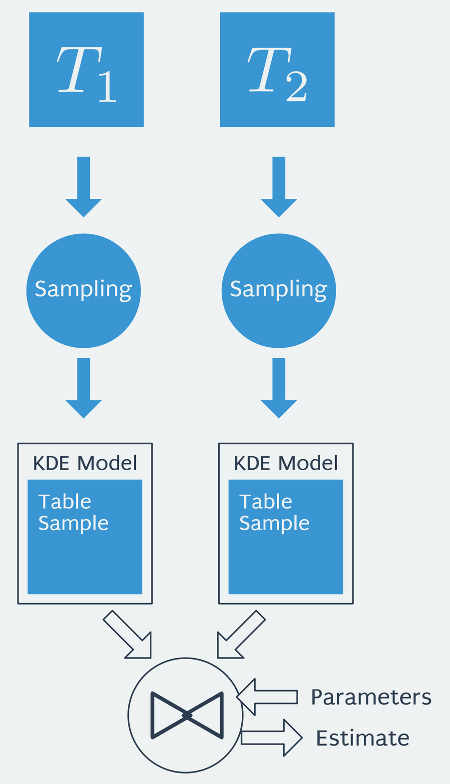


Table Model: Pruning Techniques

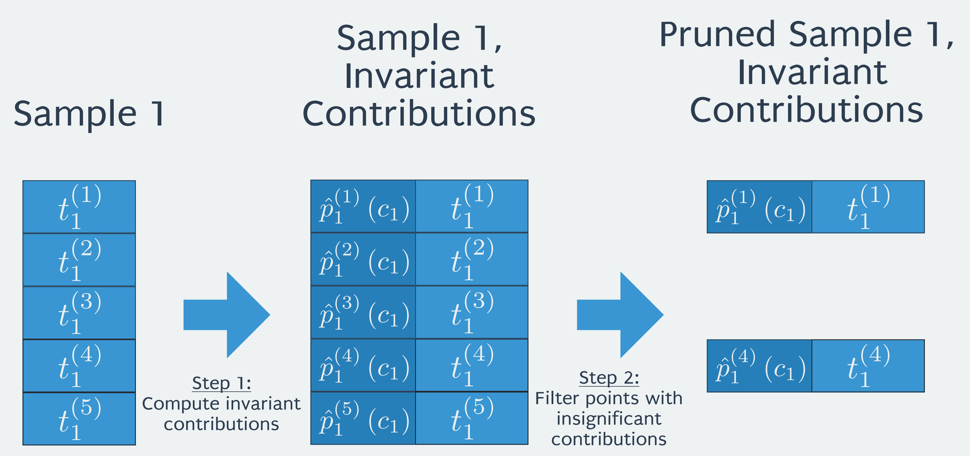
- The estimates are computed given the following equation (EQ1):

$$\frac{1}{s_1 \cdot s_2} \sum_{i=1, j=1}^{s_1, s_2} \hat{p}_1^{(i)}(c_1) \cdot \hat{p}_2^{(j)}(c_2) \cdot \hat{J}_{i,j}$$

- Sum over cross product of samples with size s_1 and s_2
- Invariant Contributions
  - Contribution of each sample point w.r.t. selection predicate
  - Depend on only one side of the cross product
- Cross Contribution
  - Join-specific contribution
  - Depends on both sides of the cross product
  - Distance function between the values on join attributes

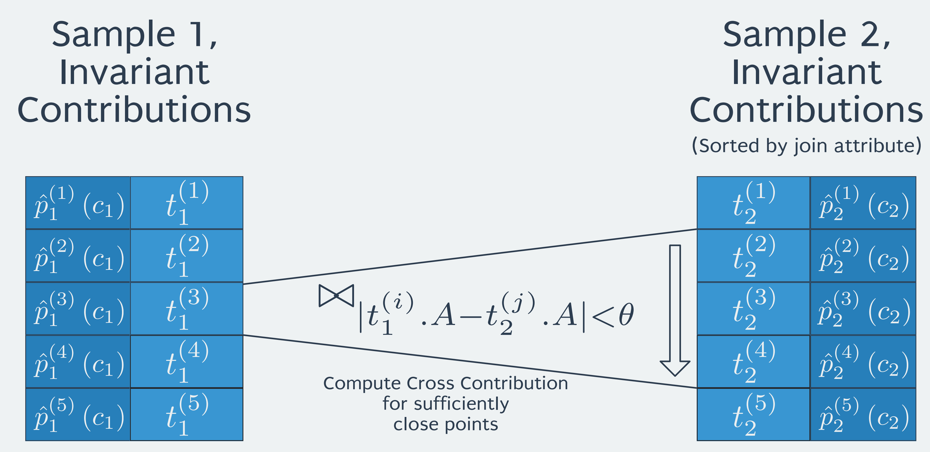
Sample Pruning

- Computes invariant contributions for every sample point
- Removes sample points with negligible contributions
- Reduces the number of input tuples to the cross product



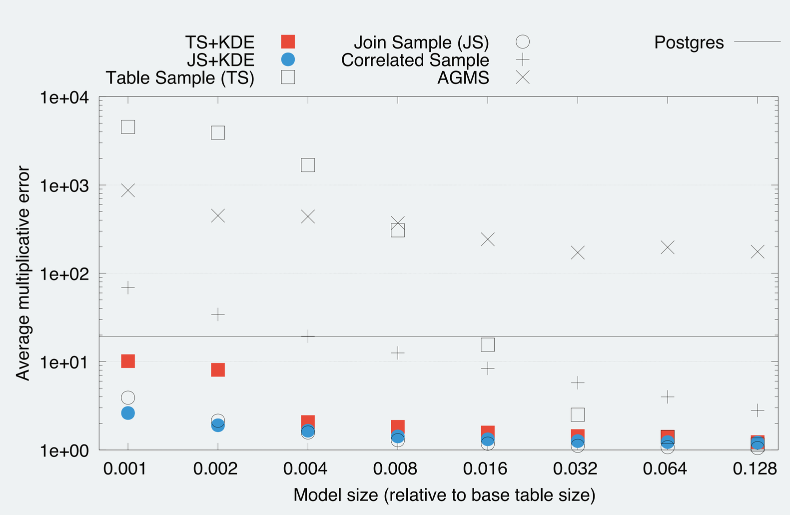
Cross Pruning

- Computes cross contribution only for sufficiently close points
- For each sample point in Sample 1 a binary search locates sufficiently close tuples in Sample 2
- Join with range predicate instead of cross product



Evaluation

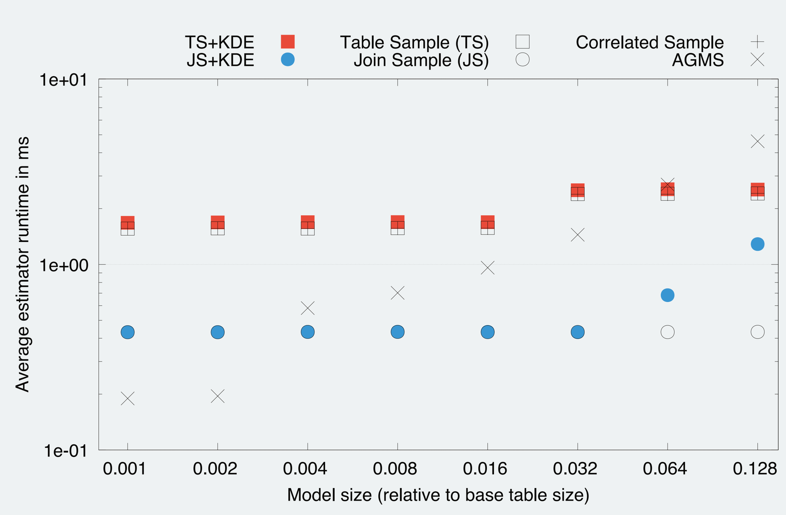
Estimation Quality



(DMV Q1 Uniform)

- Baselines
  - Postgres: 1D Statistics + Independence Assumption
  - AGMS: Sketch-based approach
  - Table/Join Sample: Uniform sample evaluation
  - Correlated Sample: Nonuniform sample evaluation
- KDE-based estimators trained on 100 workload queries
- KDE-based estimators
  - ... outperform plain samples for smaller models
  - ... converge with plain samples for larger models
  - ... outperform other baselines in most cases

Estimator Runtime



(IMDB Q1 Distinct)

- Setup
  - GPU implementations of all estimators
  - NVIDIA GTX 980
  - OpenCL
- TS+KDE runtime increase is subquadratic
  - Pruning techniques are effective
- Framework overhead dominates for smaller models
- Overhead for kernels can become significant
  - Here: JS+KDE for sizes 0.064, 0.128

Author Affiliations

TU Berlin · DFKI · Database Systems and Information Management · snowflake

Contact: martin.kiefer@tu-berlin.de

Source & Data on GitHub: martinkiefer/join-kde

Funding

Bundesministerium für Bildung und Forschung

The work received funding through the EU Horizon2020 project SAGE (671500) and from the German Ministry for Education and Research as Berlin Big Data Center BBDC (01IS14013A) and Software Campus (01IS17052).