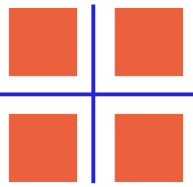
The VC-Dimension of SQL Queries and Selectivity Estimation Through Sampling



Matteo Riondato, *Mert Akdere, Uğur Çetintemel, Stanley B. Zdonik, Eli Upfal*

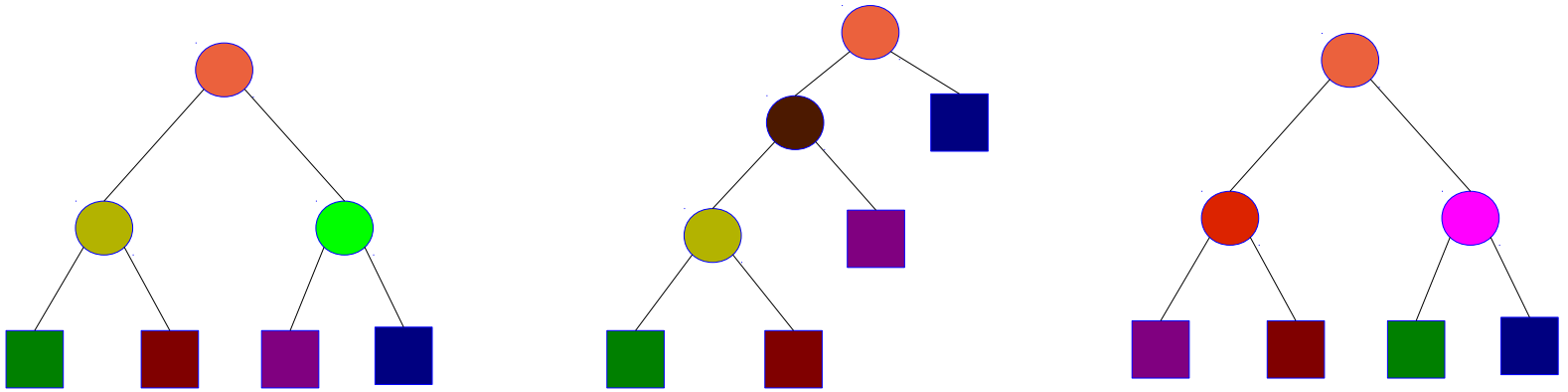
Brown University

ECML PKDD

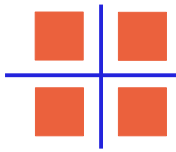
September 7th 2011

Introduction

- *Complex database queries* are composed of *elementary operations* (*select, join, ...*)
- Operations can be organized in many ways (*execution plans*) with the same final output



- The DB must choose an execution plan with the *shortest execution time*
- Not easy (*NP-hard*) ⟶ Must use *heuristics*



Selectivity

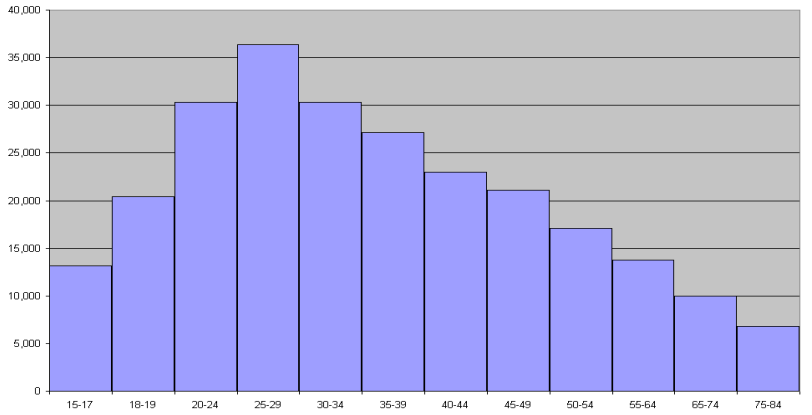
- *Heuristic:* execute *small-selectivity* operations first
- *Selectivity of* q: ratio between input and output of q

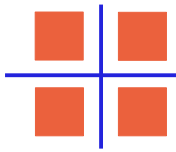
$$\sigma_{\mathrm{DB}}(q) = \frac{\#\text{ of rows in the output of } q}{\#\text{ of rows in the input of } q}$$

- *Exact quantities* not available before executing q
- *Good estimations* of $\sigma_{\mathrm{DB}}(q)$ are *necessary*

Histograms

- *Histograms*: *standard approach* to estimate $\sigma_{\mathrm{DB}}(q)$
- Approximation of *value frequency distribution* in a *single* column
- Very *fast* to build and query
- *Only work well* if frequency distributions are independent and uniform
  - Can lead to bad estimates





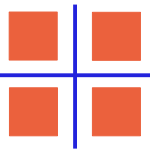
Our Goal

- Obtaining *good estimates* for all queries in a class Q
- Our estimate $\tilde{\sigma}(q)$ will be an (ε, δ)*-estimator* for $\sigma_{\mathrm{DB}}(q)$
- Error probability must hold *simultaneously* $\forall q \in Q$:

$$\Pr(\exists q \in Q \ : \ |\sigma_{\mathrm{DB}}(q) - \tilde{\sigma}(q)| > \varepsilon) < \delta$$

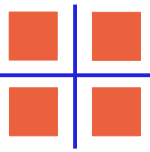
Our Approach

- High level idea:
  1) Create a *random sample* S of DB
  2) Run q on S to *obtain* $\sigma_S(q)$
  3) Use $\sigma_S(q)$ as an *estimator* for $\sigma_{\mathrm{DB}}(q)$
- Our innovations are:
  - *Guarantees* on the *quality of the estimations*
  - *Sample properties*
  - *Techniques to prove results.*



Sample Properties

- We *require* S to be:
- such that $\Pr(\exists q \in Q \ : \ |\sigma_{\mathrm{DB}}(q) - \tilde{\sigma}(q)| > \varepsilon) < \delta$
- *small,* to fit in main memory (fast query execution)
- *static*, to evaluate $\sigma_{\mathrm{DB}}(q)$ for a sequence of $q \in Q$
- $|S|$ to *depend only* on Q, not on DB
- Key question: *how large should S be?*



Major difficulty

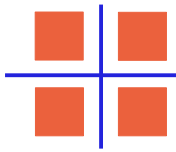
- We could use *Chernoff bounds* on the deviation $|\sigma_{\mathrm{DB}}(q) - \sigma_S(q)|$ for a *single query* $q \in Q$ [Haas96]
- *Expensive*: requires *new sample* for each query
- Cannot use *union bound* to get uniform guarantees:
  - *impractical* for long sequences of queries

Our working solution

- Avoid union bound
- Use results on *VC-Dimension* to compute $|S|$ such that for a sample S of size $|S|$, we have:

$$\Pr(\exists q \in Q \ : \ |\sigma_{\mathrm{DB}}(q) - \sigma_S(q)| > \varepsilon) < \delta$$

- A bound on the *VC-Dimension of* Q implies a bound on $|S|$
- We need a bound to the VC-Dimension of Q



VC-Dimension

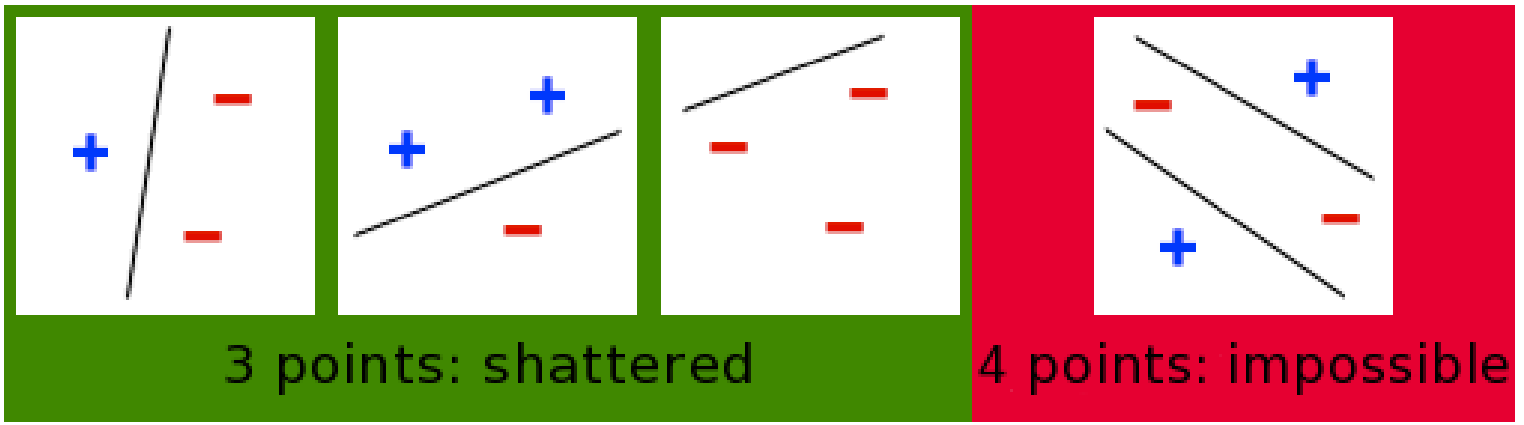
- Tool from *statistical learning theory*
- Gives a *bound to the sample size* needed to *approximately learn a function* from a given class
- Find applications in Machine Learning, Computational Geometry, ... and now Databases!

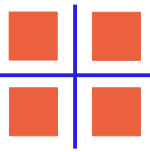
VC-Dimension

- X set of points, $F \subseteq 2^X$(*ranges*)
- The *VC-Dimension* of the *range space* (X, F) is the *cardinality* of the *largest* $A \subseteq X$ such that

$$\{r \cap A \ : \ r \in F\} = 2^A$$

- A is said to be *shattered* by F





ε-Approximation

- Fix $0 < \varepsilon, \delta < 1$
- If (X, F) has VC-dimension $\leq d$, then with probability $\geq 1 - \delta$ a *random sample* $S \subseteq X$ with size

$$|S| \geq \frac{1}{2\varepsilon^2}(d + \log\frac{1}{\delta})$$

is such that

$$\left|\frac{|f|}{|X|} - \frac{|S \cap f|}{|S|}\right| \leq \varepsilon, \forall f \in F$$

- Such a sample S is called an *ε-approximation*

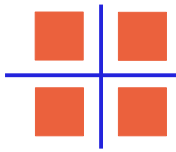
How does it help us?

- In our case:
  - $X = \mathrm{DB}$ (Cartesian product of tables in DB)
  - $F_Q = \{\text{output of } q, \forall q \in Q\}$
- If $S \subseteq \mathrm{DB}$ is an ε-*approximation* for (DB, F_Q):

$$\left| \frac{|f_q|}{|X|} - \frac{|S \cap f_q|}{|S|} \right| \leq \varepsilon, \forall f_q \in F_Q$$

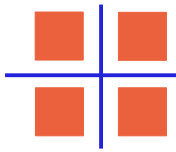
$\sigma_{\mathrm{DB}}(q)$ $\qquad$ $\sigma_S(q)$

i.e. $\boxed{|\sigma_{\mathrm{DB}}(q) - \sigma_S(q)| \leq \varepsilon, \forall q \in Q}$



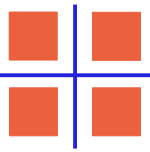
Our Results

- We give a *bound to the VC-Dimension of queries*
- We characterize queries by their *SQL expression*
- The *more complex* the expression, the *higher the VC-dimension* of the corresponding class
- The VC-dimension is *completely described* by the SQL expression
- It only depends on Q and not on the data in DB



Our Results

- Start from *simple selection queries*, and build up from there (*multiple conditions*, *join queries*, ...)
- Use previous results from *computational geometry*
- Develop new direct bounds to the VC-dimension.
- Some *technical details* (Full paper on arXiv)

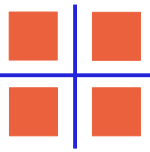


Generic Queries

- Let
  - k: maximum number of join operations in a query
  - b: maximum number of conditions in a selection
  - m: maximum number of columns in a table
- $Q_{k,b,m}$: class including all such queries
- We have:

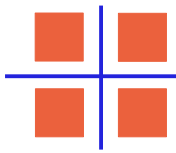
$$VC(X, Q_{k,b,m}) = \tilde{O}(k^2 mb)$$

and obtain the sample size for a ε-approximation



Sampling Issues

- The theorem requires a *sample of the Cartesian product* of the tables
- It is *expensive* and *not practical* to compute and store the Cartesian product
- We want to *keep the table layout* in the sample

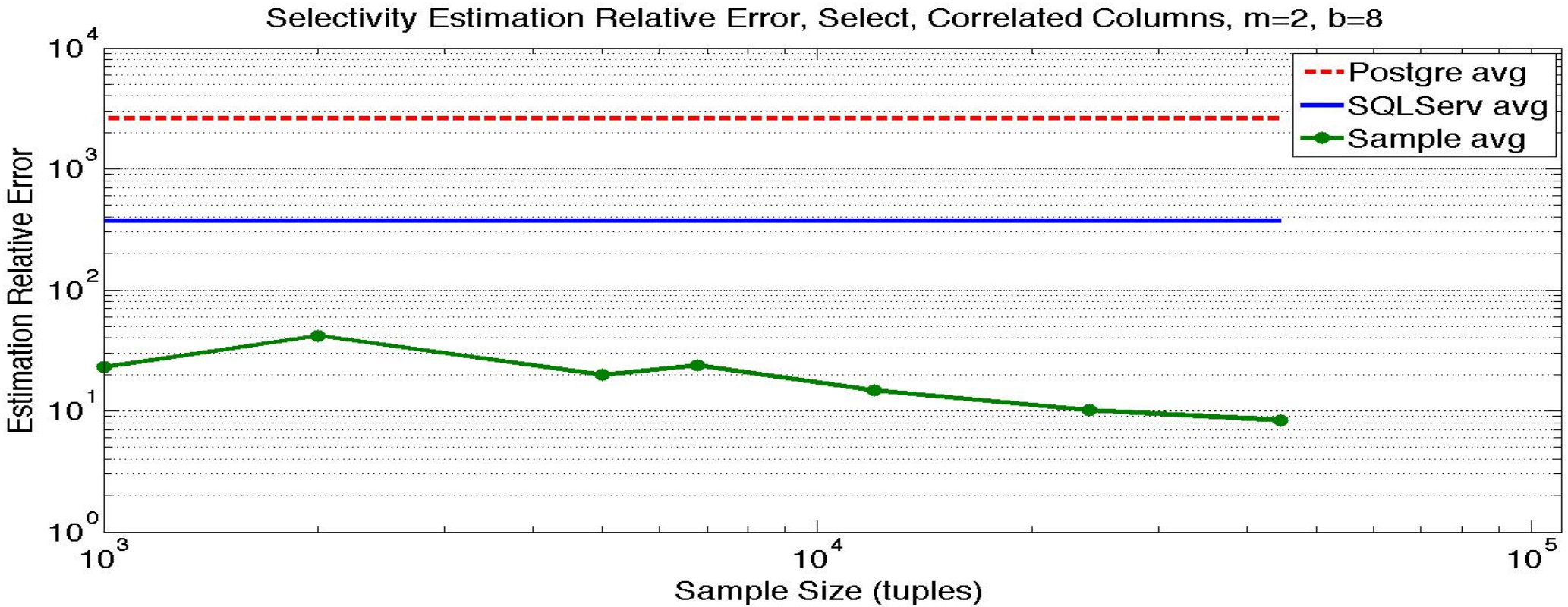


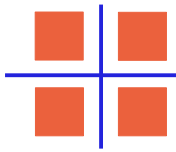
Solution

- Sample *independently* $|S|$ tuples from each table
- Assign index $i \in [1, |S|]$ to i-th sampled tuple
- Combinations of tuples with *same index* are members of the sample S
- Only these contribute to output size
- Need *one scan* of the output to identify them

Experiments

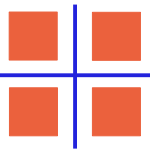
- Sample can fit into main memory
- Estimations are *always* within ε from real value
- We *outperform* PostgreSQL and MS SQLServer by *orders of magnitude*





Conclusion

We use a very *theoretical tool* from *statistical learning theory* (*VC-Dimension*) to *efficiently* solve a very important *practical problem* in *databases* (*selectivity estimation*)



Thank you!

- Questions?

matteo@cs.brown.edu

Simple Selection Queries

- T: table with m columns
- X: the tuples of T as points of $\mathbb{R}^m$
- F: queries in the form SELECT * FROM T WHERE T_i op v
  - op is ≤ or ≥, $v \in D(T_i)$, the domain of T_i
  - *single open interval on single column*
- $f \in F$ can be seen as the *half space*

$$\{(x_1, \ldots, x_i, \ldots, x_m) : x_j \in D(T_j) \text{ for } j \neq i \text{ and } x_i \text{ op } v\}$$

- Known result from geometry:

$$VC(X, F) < VC(\mathbb{R}^m, H) = m + 1$$

Half Spaces in $\mathbb{R}^m$

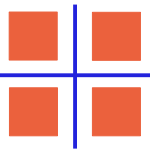
General Selection Queries

- SELECT * FROM T WHERE

$$T_{j_1}\ \mathrm{op}_1\ v_1\ \mathrm{bool}_1 \ldots \mathrm{bool}_{h-1}\ T_{j_h}\ \mathrm{op}_h\ v_h$$

- bool_i: either AND or OR
- Output: *combination of outputs* of queries from F
  - one query for each condition $T_{j_i}\ \mathrm{op}_i\ v_i$
  - combined with *union for ORs*, *intersection for ANDs*
- F_h = queries involving at most h members of F
- *Technical lemma* on unions and intersections:

$$VC(X, F_h) \leq 3(m+1)h\log((m+1)h)$$



Join Queries

- T_1, T_2 tables, $T_i.C$ column in T_i
- SELECT * FROM T_1, T_2 WHERE

$$T_1.C = T_2.C \text{ AND } sel_1 \text{ AND } sel_2$$

  - sel_i selection query on T_i
- Output of join depends on outputs of sel_1, sel_2
- VC-Dimension of join queries depends on VC-Dimension of selection queries on T_1, T_2
- Makes the computation of the VC-Dimension of joins more difficult

Join Queries

- $X = T_1 \times T_2$, F_i: family of selection queries on T_i
- $v_i = VC(X, F_i)$
- F_C : outputs of join queries between T_1, T_2 along column C when F_1, F_2 are allowed on T_1, T_2
- $VC(X, F_C) \leq 3(v_1 + v_2)\log(v_1 + v_2)$
- Generalizing to queries with up to k join operations

$$VC(X, Q_k) \leq 4k(\sum_{i=1}^{k} v_i)\log(k\sum_{i=1}^{k} v_i)$$

Outline of the proof

- $A \subseteq X, |A| = v$
- $A_i = \{x \in T_i : (x, y) \in A, y \in T_j\}, i = 1, 2, j = 2, 1$
- $P_{F_C}(A) \subseteq P_{F_1}(A_1) \times P_{F_2}(A_2)$
- $|P_F C(A)| \leq |P_{F_1}(A_1) \times P_{F_2}(A_2)| \leq v^{v_1+v_2}$
- if $v^{v_1+v_2} < 2^v$ then A can not be shattered
- This happens if $v > 3(v_1 + v_2)\log(v_1 + v_2)$.