Accurate Summary-based Cardinality Estimation Through the Lens of Cardinality Estimation Graphs

Jeremy Chen
University of Waterloo
jeremy.chen@uwaterloo.ca

Yuqing Huang
University of Waterloo
y558huan@uwaterloo.ca

Mushi Wang
University of Waterloo
m358wang@uwaterloo.ca

Semih Salihoglu
University of Waterloo
semih.salihoglu
@uwaterloo.ca

Kenneth Salem
University of Waterloo
ken.salem@uwaterloo.ca

ABSTRACT

We study two classes of summary-based cardinality estimators that use statistics about input relations and small-size joins: (i) optimistic estimators, which were defined in the context of graph database management systems, that make uniformity and conditional independence assumptions; and (ii) the recent pessimistic estimators that use information theoretic linear programs (LPs). We show that optimistic estimators can be modeled as picking bottom-to-top paths in a *cardinality estimation graph* (CEG), which contains subqueries as nodes and edges whose weights are average degree statistics. We show that existing optimistic estimators have either undefined or fixed choices for picking CEG paths as their estimates and ignore alternative choices. Instead, we outline a space of optimistic estimators to make an estimate on CEGs, which subsumes existing estimators. We show, using an extensive empirical analysis, that effective paths depend on the structure of the queries. We next show that optimistic estimators and seemingly disparate LP-based pessimistic estimators are in fact connected. Specifically, we show that CEGs can also model some recent pessimistic estimators. This connection allows us to provide insights into the pessimistic estimators, such as showing that they have combinatorial solutions.

1. INTRODUCTION

The problem of estimating the output size of a natural multi-join query (henceforth *join query*), is a fundamental problem that is solved in the query optimizers of database management systems to generate efficient query plans. This problem arises both in relational systems as well as those that manage graph-structured data where systems need to estimate the cardinalities of subgraphs in their input graphs. It is well known that both problems are equivalent, since subgraph queries can equivalently be written as join queries over binary relations that store the edges of a graph.

Perhaps the most widely adopted estimators are *summary-based*, in which the DBMS uses statistics about a database in a procedure to derive an estimate. In this paper, we study two classes of summary-based techniques, which we will model through a common framework:

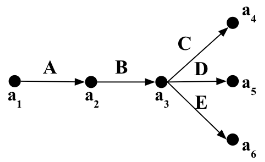


Figure 1: Example subgraph query Q_{5f}.

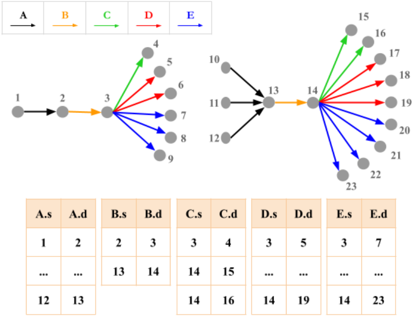
- *Optimistic estimators* [2, 22, 24, 27] from graph-based systems store as statistics the cardinalities of small size joins/subgraphs, and generate estimates by combining those statistics using algebraic formulas that make independence and uniformity assumptions. We refer to these as "optimistic", since they may underestimate the true cardinalities of queries.
- *Pessimistic estimators* [16] [6], which were introduced in the context of relational systems. Pessimistic estimators store as statistics the *degrees* of the values in columns, i.e., the number of times a value appears, and use linear programs (LPs) and have the guarantee that the estimate is an upper bound on the true cardinality of a query.

To motivate our first contribution, consider a database that consists of an edge-labeled graph with capital letter labels A, B, C, etc. This can be modeled as a set of binary relations, such as `A(src, dst)`. Consider the "subgraph query" in Figure 1, asking for matches of a 5-edge subgraph, which is equivalent to a join of 5 relations. Given the accurate cardinalities of all subqueries of size ≤ 2 there are 252 formulas to make an estimate. Two examples are:

- $|\xrightarrow{A}\xrightarrow{B}| \times \frac{|\xrightarrow{B}\xrightarrow{C}|}{|\xrightarrow{B}|} \times \frac{|\xleftarrow{C}\xrightarrow{D}|}{|\xrightarrow{C}|} \times \frac{|\xleftarrow{D}\xrightarrow{E}|}{|\xrightarrow{D}|}$
- $|\xrightarrow{A}\xrightarrow{B}| \times \frac{|\xrightarrow{B}\xrightarrow{D}|}{|\xrightarrow{B}|} \times \frac{|\xleftarrow{C}\xrightarrow{D}|}{|\xrightarrow{D}|} \times \frac{|\xrightarrow{B}\xrightarrow{E}|}{|\xrightarrow{B}|}$

In previous work [2, 22, 24], the choice of which of these estimates to use has either been unspecified or fixed without acknowledging possible other choices. Our first contribution aims to answer the following questions: (1) What is the space of estimation formulas for a qiven a query? (2) Which formulas should be picked to make more accurate estimates?

We begin by showing that the algebraic formulas of prior



A.s	A.d	B.s	B.d	C.s	C.d	D.s	D.d	E.s	E.d
1	2	2	3	3	4	3	5	3	7
...	...	13	14	14	15	...	...	...	...
12	13			14	16	14	19	14	23

Figure 2: Example dataset in graph and relational formats.

optimistic estimators can be modeled as picking a bottom-to-top path in a weighted *cardinality estimation graph* (CEG), which we call CEG_O, for **O**ptimistic. In this CEG, nodes are intermediate sub-queries and edges weights are average degree statistics that extend sub-queries to larger queries. For example, the CEG_O for the query in Figure 1 and the dataset in Figure 2 is shown in Figure 3. Each path of this CEG corresponds to a possible formula where the estimate is the product of the weights of the edges in the path.

We describe a space of 9 estimators, defined by different choices in picking a CEG path for making an optimistic estimate. This space subsumes and extends the choices made by existing optimistic estimators. We then empirically show that the better performing optimistic estimator in this CEG space depends on the structure of the query. On acyclic queries and queries with small-size cycles, we advise the use the *maximum-weight paths* because the use independence assumptions tend to underestimate the true cardinalities of queries, and this can be offset by picking the highest estimating formula. In contrast, on queries that contain larger cycles, optimistic estimators estimate the number of paths, rather than cycles. Here we advise the use of minimum-weight paths, as real-world graphs contain many more paths than cycles.

As our next main contribution, we show that CEGs are expressive enough to model also the recent LP-based pessimistic estimators. Specifically, we show that we can replace the edge weights of CEG_O (which are average degrees) with maximum degrees of base relations and small-size joins, and construct a new CEG, which we call CEG_M. Unlike the optimistic estimators, where the choice of path is not clear, we now show that picking the minimum weight path would (provably) be the most accurate estimate and this path is indeed equivalent to the solution of the LP that defines a pessimistic estimator, called MOLP [16]. We therefore show that *both subgraph summary-based optimistic estimators and the recent LP-based pessimistic ones, which were not known to be related, can be seen as different instances of a broader class of estimators that pick paths through CEGs.* Modeling these two classes of estimators in a common framework has several benefits that we demonstrate, e.g., one can apply optimizations for one class to the other. In addition, our paper opens the possibility of defining many other CEGs that use different statistics as edge weights (e.g., entropies of certain columns). We hope future work can define and evaluate the accuracies of such CEGs.

For readers interested in the theory of pessimistic estimators, we note that CEGs are also very useful mathematical tools to prove properties of pessimistic estimators. For example, using CEGs in our proofs, we can derive combinatorial proofs to some properties of MOLP, e.g., that MOLP is identical to the pessimistic estimator proposed by Cai et al [6] on acyclic queries over binary relations.

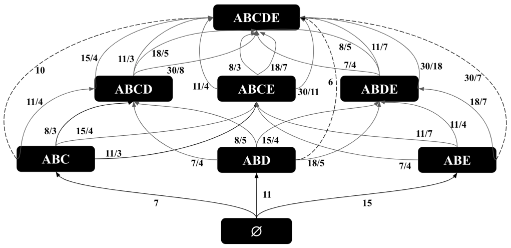


Figure 3: CEG_O for query Q_{5f} in Figure 1 when the Markov table (§4) contains joins up to size 3.

This paper is based on an earlier one [9]. This version aims to make the contents of our previous paper more accessible to the general database audience. We have shortened or removed discussion of several experiments and applications of CEGs, and we refer readers to the earlier paper for that material.

2. QUERY AND DATABASE NOTATION

We consider conjunctive queries of the form

$$Q(\mathcal{A}) = R_1(\mathcal{A}_1), \ldots, R_m(\mathcal{A}_m)$$

where $R_i(\mathcal{A}_i)$ is a relation with attributes $\mathcal{A}_i$ and $\mathcal{A} = \cup_i \mathcal{A}_i$. Most of the examples used in this paper involve edge-labeled subgraph queries, in which case each R_i is modeled a binary relation containing a subset of the edges in a graph as source/destination pairs. Note that even if these edges contain other properties that can be modeled as additional columns, for the purpose of the join query, they are not relevant. Figure 2 presents an example showing a graph with edge labels A, B, C, D, and E. This graph can be represented using five binary relations, one for each edge label, as shown in Figure 2. We will often represent queries over such relations using a graph notation. For example, consider the relations A and B from Figure 2. We will represent the query $Q(a_1, a_2, a_3) = A(a_1, a_2) \bowtie B(a_2, a_3)$ as $a_1 \xrightarrow{A} a_2 \xrightarrow{B} a_3$. Similarly, the query $Q(a_1, a_2, a_3) = A(a_1, a_2) \bowtie B(a_3, a_2)$ will be represented as $a_1 \xrightarrow{A} a_2 \xleftarrow{B} a_3$.

Note on binary vs arbitrary relations: We emphasize that CEGs, which will be introduced in Section 3, do not require relations to be binary. In fact the CEG that we will define for pessimistic estimators in Section 5 assumes relations with arbitrary arities. However, our empirical evaluation on optimistic estimators uses binary relations. This is because optimistic estimators have - for very practical reasons - been developed in the context of graph-based systems, where joins are over binary relations. The reason is that optimistic estimators store as statistics the sizes of all possible small-size joins, and when relations are binary it is practical for a system to know and store the set of possible small-size joins in advance. In contrast, in RDBMSs that store relations with arbitrary number of columns and where columns can be arbitrarily joined with each other, the number of possible small-size joins can be very large.

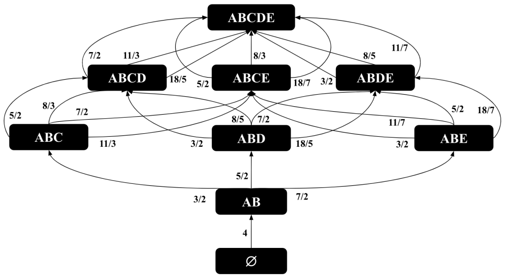


Figure 4: CEG_O for query Q_{5f} from Figure 1 when the Markov table (§4) contains joins up to size 2.

3. CEG OVERVIEW

Next, we offer some intuition for *cardinality estimation graphs* (CEGs). A CEG for a query Q consists of:

- Vertices labeled with subqueries of Q, where subqueries are defined by subsets of Q's relations or attributes.
- Edges from smaller subqueries to larger subqueries, labeled with *extension rates* which represent the cardinality of the larger subquery relative to that of the smaller subquery.

Each bottom-to-top path (from $\emptyset$ to Q) in a CEG represents a different way of generating a cardinality estimate for Q. An estimator using a CEG picks one of these paths as an estimate. The estimate of a path is the product of the extension rates along the edges of the path. Equivalently one can put the logarithms of the extension rates as edge weights and sum the logarithms.

Figure 4 illustrates a CEG[1] for the query Q_{5f} shown in Figure 1 over the relations shown in Figure 2, assuming that statistics are available for any size-2 subqueries of Q_{5f}. Depending on the semantics of the edges in a CEG, there can be multiple edges between two sub-queries in a CEG. For example the last components of the two formulas, we present in Section 1 for Q_{5f}, $|\xleftarrow{D}\xrightarrow{E}|/|\xrightarrow{D}|$ and $|\xrightarrow{B}\xrightarrow{E}|/|\xrightarrow{B}|$, correspond to parallel edges that extend the sub-query over $ABCD$ edges with an E edge to Q_{5f}. Consider the leftmost path. The first extension rate from $\emptyset$ to $a_1 \xrightarrow{A} a_2 \xrightarrow{B} a_3$ is the known cardinality of $a_1 \xrightarrow{A} a_2 \xrightarrow{B} a_3$, which is 4, and the second extension rate has a weight 3/2, intuitively estimating that each $a_1 \xrightarrow{A} a_2 \xrightarrow{B} a_3$ path will extend to 3/2 many $a_1 \xrightarrow{A} a_2 \xrightarrow{B} a_3 \xrightarrow{C} a_4$ paths. Continuing the extensions, the final estimate is $4 \times \frac{3}{2} \times \frac{5}{2} \times \frac{7}{2} = 52.5$.

In the rest of this paper, we will show how some prior optimistic and pessimistic estimators can be modeled as instances of this generic estimator using different CEGs.

4. OPTIMISTIC ESTIMATORS

The estimators that we refer to as *optimistic* use statistics about the input database in formulas that make uniformity and independence or conditional independence assumptions. We focus on three estimators: *Markov tables* [2] from XML databases, graph summaries [22] from RDF databases, and the graph catalogue estimator of the Graphflow system [24] for managing property graphs. As we explain momentarily, despite being designed for systems that adopt different graph-based data models, these estimators are all extensions of each other.

Table 1: Example Markov table for h=2.

Path	|Path|
$\xrightarrow{B}$	2
$\xrightarrow{A}\xrightarrow{B}$	4
$\xrightarrow{B}\xrightarrow{C}$	3
...	...

We begin by giving an overview of the Markov tables estimator [2]. A Markov table of length $h \geq 2$ stores the cardinality of each path in an XML document's element tree up to length h and uses these to make predictions for the cardinalities of longer paths. Table 1 shows a subset of the entries in an example Markov table for $h = 2$ for our running example dataset from Figure 2. The formula to estimate a 3-path using a Markov table with $h = 2$ is to multiply the cardinality of one of the 2-paths with the consecutive 2-path divided by the cardinality of the common edge. For example, consider the query $Q_{3p} = \xrightarrow{A}\xrightarrow{B}\xrightarrow{C}$ against the dataset in Figure 2. The formula for Q_{3p} would be: $|\xrightarrow{A}\xrightarrow{B}| \times (|\xrightarrow{B}\xrightarrow{C}|/|\xrightarrow{B}|)$. The formula assumes that the number of C edges that each B edge extends to is uniformly $r = |\xrightarrow{B}\xrightarrow{C}|/|\xrightarrow{B}|$. Equivalently, this is the "average C-degree" of nodes in the $\xrightarrow{B}\xrightarrow{C}$ paths. The result of this formula is $4 \times \frac{3}{2} = 6$, which underestimates the true cardinality of 7. The graph summaries [22] for RDF databases and the graph catalogue estimator [24] have extended the contents of what is stored in Markov tables, respectively, to other acyclic joins, such as stars, and small cycles, such as triangles.

4.1 Space of Possible Optimistic Estimators

We next represent optimistic estimators using a CEG that we call CEG_O. We assume that the given query Q is connected. CEG_O consists of the following:

- *Vertices:* For each connected subset of relations $S \subseteq \mathcal{R}$ of Q, we have a vertex in CEG_O with label S. This represents the sub-query $\bowtie_{R_i \in S} R_i$.
- *Edges:* Consider two vertices with labels S and S' s.t., $S \subset S'$. Let $\mathcal{D}$=$S' \setminus S$ (for **d**ifference), and $\mathcal{E} \supset \mathcal{D}$ (for **e**xtension) be a Markov table entry, and $\mathcal{I}$=$\mathcal{E} \cap S$ (for **i**ntersection). If $\mathcal{E}$ and $\mathcal{I}$ exist in the Markov table, then there is an edge with weight $\frac{|\mathcal{E}|}{|\mathcal{I}|}$ from S to S' in CEG_O.

This gives the core structure of CEG_O though when making estimates there are several simple rules that remove some edges in CEG_O from prior work that limit the paths considered in CEG_O (see the original version of our paper [9]). In general, there may be multiple $(\emptyset, Q)$ paths that lead to different estimates in CEG_O:

Example 1: Consider the CEG_O for Q_{5f} shown in Figure 4 which uses a Markov table of size 2. There are 36 $(\emptyset, Q)$ paths leading to 7 different estimates. Two examples are:

- $|\xrightarrow{A}\xrightarrow{B}| \times \frac{|\xrightarrow{B}\xrightarrow{C}|}{|\xrightarrow{B}|} \times \frac{|\xrightarrow{B}\xrightarrow{D}|}{|\xrightarrow{B}|} \times \frac{|\xrightarrow{B}\xrightarrow{E}|}{|\xrightarrow{B}|} = 52.5$
- $|\xrightarrow{A}\xrightarrow{B}| \times \frac{|\xrightarrow{B}\xrightarrow{C}|}{|\xrightarrow{B}|} \times \frac{|\xleftarrow{C}\xrightarrow{D}|}{|\xrightarrow{C}|} \times \frac{|\xleftarrow{D}\xrightarrow{E}|}{|\xrightarrow{D}|} = 57.6$

Example 2: Similarly, consider estimating Q_{5f} now with a Markov table with up to 3-size joins. The new CEG_O is

[1] Specifically, it is a CEG_O, defined in Section 4.

shown in Figure 3, which contains multiple paths leading to 2 different estimates:

- $|\xrightarrow{A}\xrightarrow{B}\xrightarrow{C}| \times \frac{|\xleftarrow{C}\underset{E}{\overset{D}{\rightrightarrows}}|}{|\xrightarrow{C}|}$
- $|\xrightarrow{A}\xrightarrow{B}\xrightarrow{C}| \times \frac{|\xrightarrow{A}\xrightarrow{B}\xrightarrow{D}|}{|\xrightarrow{A}\xrightarrow{B}|} \times \frac{|\xrightarrow{A}\xrightarrow{B}\xrightarrow{E}|}{|\xrightarrow{A}\xrightarrow{B}|}$

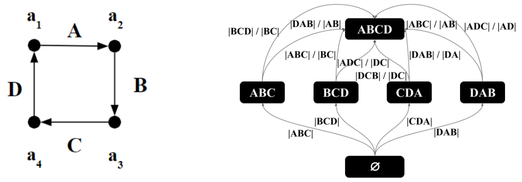
Both formulas start by $|\xrightarrow{A}\xrightarrow{B}\xrightarrow{C}|$. The first "short-hop" formula makes one fewer independence assumption than the "long-hop" formula, which is an advantage. In contrast, the first estimate also makes a uniformity assumption that conditions on a smaller-size join, which might make it less accurate than the two assumptions made in the long-hop estimate, which condition on 2-size joins.

Any optimistic estimator implementation needs to make choices about which formulas to use, which corresponds to picking paths in CEG_O. We systematically identify a space of choices that an optimistic estimator can make along two parameters that capture the choices made in prior work:

- *Path length:* The estimator can identify a set of paths to consider based on the path lengths, i.e., number of edges or hops, in CEG_O, which can be: (i) maximum-hop (`max-hop`); (ii) minimum-hop (`min-hop`); or (iii) any number of hops (`all-hops`). Let $\mathcal{P}$ be the set of paths an estimator picks.
- *Aggregator:* To derive a final estimate, the estimator has to aggregate the estimates in $\mathcal{P}$. We identify three aggregators: (i) the path with the largest estimate (`max-aggr`); (ii) the path with the lowest estimate (`min-aggr`); or (iii) the average of all the estimates in $\mathcal{P}$ (`avg-aggr`).

Any combination of these two choices can be used to design an optimistic estimator. The original Markov tables [2] chose the `max-hop` paths. In reference [2] queries were paths, so when the path length is chosen, any CEG_O path gives the same estimate. Therefore an aggregator is not needed. Graph summaries [22] chooses the `min-hop` paths and leaves the aggregator unspecified. Graph catalogue [24] picks the `min-hop` and `min-aggr` aggregator. None of these estimators consider alternative choices an estimator can make. Instead, we do a systematic experimental analysis of this space of estimators in Section 6 and show that the best choices depend on the query structure, as optimistic estimators behave differently on different structures: for acyclic queries and queries with small cycles these estimators tend to underestimate and for queries with larger cycles, they tend to overestimate. We next make an observation to explain this difference.

Recall that a Markov table stores the cardinalities of patterns up to some size h. Given a Markov table with $h \geq 2$, optimistic estimators can produce estimates for any acyclic query with size larger than h. However, faced with a large cyclic query Q, optimistic estimators do not actually produce estimates for Q. Instead, they produce an estimate for a similar acyclic Q' that includes all of Q's edges but is not closed. To see this, consider the following example:



(a) 4-cycle query. (b) CEG_O.

Figure 5: A 4-cycle query and its CEG_O.

Example 1. *Consier a 4-cycle query in Figure 5a using a Markov table with* h=3. *The* CEG_O *for this setting is shown in Figure 5b. Consider the left most path corresponding to the formula:* $|\xrightarrow{A}\xrightarrow{B}\xrightarrow{C}| \times |\xrightarrow{B}\xrightarrow{C}\xrightarrow{D}| / |\xrightarrow{B}\xrightarrow{C}|$. *This formula is in fact estimating a 4-path* $\xrightarrow{A}\xrightarrow{B}\xrightarrow{C}\xrightarrow{D}$ *rather than the 4-cycle. This is true for each path in* CEG_O.

More generally, when queries contain cycles of length $> h$, CEG_O breaks cycles in queries into paths. Therefore, estimates over CEG_O can lead to *over*estimates for queries with large cycles, as there are often significantly more paths than cycles in real-world graphs.

5. PESSIMISTIC ESTIMATORS

Starting from the seminal result by Atserias, Grohe, and Marx in 2008 [5], several upper bounds have been provided for the output sizes of join queries under different known statistics. For example the initial upper bound from reference [5], now called the *AGM bound*, used only the cardinalities of each relation, while later bounds, DBPLP [16], MOLP [16], and CLLP [1] used maximum degrees of the values in the columns and improved the AGM bound. Since these bounds are upper bounds on the query size, they can be used as *pessimistic estimators.* This was done recently by Cai et al. [6] in an actual estimator implementation. We refer to this as the CBS estimator, after the names of the authors. We next show that some of the recent pessimistic estimators [16, 6] can also be modeled as making an estimate using a CEG.

5.1 MOLP

MOLP was defined in reference [16] as a tighter bound than the AGM bound that uses additional degree statistics about input relations that AGM bound does not use. We first review the formal notion of a degree. Let $\mathcal{X}$ be a subset of the attributes $\mathcal{A}_i$ of some relation $\mathcal{R}_i$, and let v be a possible value of $\mathcal{X}$. The *degree* of v in $\mathcal{R}_i$ is the number of times v occurs in $\mathcal{R}_i$, i.e. $deg(\mathcal{X}(v), \mathcal{R}_i) = |\{t \in \mathcal{R}_i | \pi_{\mathcal{X}}(t) = v\}|$. For example, in Figure 2, $deg(s(3), E) = 3$ because the outgoing E-degree of vertex 3 is 3. Similarly $deg(d(2), A)$ is 1 because the incoming A-degree of vertex 2 is 1. We also define $deg(\mathcal{X}, \mathcal{R}_i)$ to be the maximum degree in $\mathcal{R}_i$ of any value v over X, i.e., $deg(X, \mathcal{R}_i) = \max_v deg(\mathcal{X}(v), \mathcal{R}_i)$. So, $deg(d, A) = 3$ because vertex 13 has 3 incoming A edges, which is the maximum A-in-degree in the dataset. The notion of degree can be generalized to $deg(X(v), Y, \mathcal{R}_i)$, which refers to the "degree of a value v over attributes X in $\pi_Y \mathcal{R}_i$", which counts the number of times v occurs in $\pi_Y(\mathcal{R}_i)$. Similarly, we let $deg(X, Y, \mathcal{R}_i) = \max_v deg(X(v), Y, \mathcal{R}_i)$. Suppose a system has stored $deg(X, Y, \mathcal{R}_i)$ statistics for each possible $\mathcal{R}_i$ and $X \subseteq Y \subseteq \mathcal{A}_i$. MOLP is the output of this LP:

Maximize $s_{\mathcal{A}}$

$s_{\emptyset} = 0$

$s_X \leq s_Y, \forall X \subseteq Y$

$s_{Y \cup E} \leq s_{X \cup E} + \log(deg(X, Y, \mathcal{R}_i)), \forall X, Y, E \subseteq \mathcal{A}, X \subseteq Y \subseteq \mathcal{A}_i$

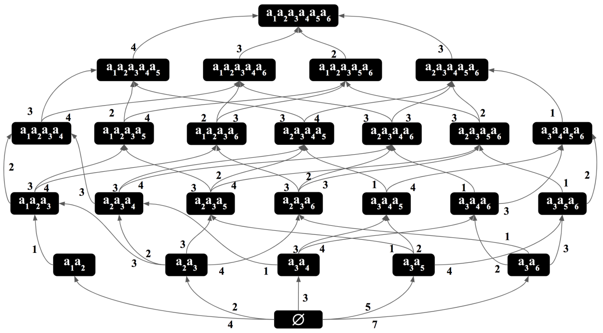


Figure 6: CEG_M for query Q_{5f} in Figure 1.

The base of the logarithm can be any constant and we take it as 2. Let m_A be the optimal value of MOLP. Reference [16] has shown that 2^{m_A} is an upper bound on the size of Q.

MOLP CEG (CEG_M): It is not easy to directly see the solution of the MOLP on our running example. However, we can represent the MOLP bound as the cost of minimum-weight $(\emptyset, Q)$ path in a CEG that we call CEG_M.

- *Vertices:* For each $X \subseteq \mathcal{A}$, the variable s_X in MOLP represents an upper bound on the size of $Q_X = \Pi_X Q$. Therefore, for each $X \subseteq \mathcal{A}$ there is a vertex in CEG_M.
- *Extension Edges:* For each $s_{Y\cup E} \leq s_{X\cup E} + \log(deg(X, Y, \mathcal{R}_i))$ inequality, there is an edge with weight $\log(deg(X, Y, \mathcal{R}_i))$ between any $W_1 = X \cup E$ and $W_2 = Y \cup E$. These inequalities intuitively indicate the following: each tuple $t_{X\cup E} \in Q_{X\cup E}$ can extend to at most $deg(X, Y, \mathcal{R}_i)$ $Q_{Y\cup E}$ tuples.
- *Projection Edges:* For each $s_X \leq s_Y$ inequality (i.e., $\forall X \subseteq Y$), add an edge with weight 0 from Y to X. These indicate that the size of Q_X is at most the size of Q_Y, if Y is a larger subquery.

Figure 6 shows the CEG_M of our running example. The figure uses actual degrees instead of their logarithms as edge weights and omits the projection edges.

Theorem 5.1. *Let Q be a query with degree statistics $deg(X, Y, R_i)$ for each $\mathcal{R}_i$ and $X \subseteq Y \subseteq \mathcal{A}_i$. The optimal solution m_A to the MOLP of Q is the weight of the minimum-weight $(\emptyset, \mathcal{A})$ path in CEG_M.*

Theorem 5.1's proof can be found in reference [8]. With this connection, readers can verify the MOLP bound in our example by inspecting the paths in Figure 6: the minimum-weight $(\emptyset, \mathcal{A})$ path has a weight of 96, corresponding to the leftmost path. We make two observations:

Observation 1: Reference [16] proves through a numeric LP-duality argument that $2^{m_{\mathcal{A}}}$ is an upper bound on the the output size of the query (OUT), i.e., $OUT \leq 2^{m_{\mathcal{A}}}$ (see Prop. 2 [16]). Our CEG formulation of MOLP provides arguably a simpler proof of this property. Note that each $(\emptyset, \mathcal{A})$ path in CEG_M corresponds to a sequence of extensions from $\emptyset$ to Q and is an estimate of the cardinality of Q. Since we are using maximum degrees on the edge weights, each $(\emptyset, \mathcal{A})$ path is by construction an upper bound on Q. Since for any $(\emptyset, \mathcal{A})$ path P in CEG_M, $OUT \leq 2^{w(P)}$ and by Theorem 5.1, $m_{\mathcal{A}}$ is equal to the weight of the minimum-weight $(\emptyset, \mathcal{A})$ path in CEG_M, $OUT \leq 2^{m_{\mathcal{A}}}$.

Observation 2: Theorem 5.1 implies that MOLP can be solved using a combinatorial shortest-path algorithm instead of a numeric LP solver.

In our original paper [9], we also review the CBS pessimistic estimator [6] and show, using CEGs in our proofs, that MOLP is as tight as CBS and is exactly the same as CBS on acyclic join queries over binary relations. Our use of CEGs as mathematical tools in our proofs may be of interest to readers interested in the theory of worst-case bounds on the sizes of join queries. Overall, our results show that CEGs are expressive enough to model two prior pessimistic estimators as well. One benefit of this is that we can apply optimizations done in one class of estimators to the other. In the longer version of our paper, we show an example of this by applying an optimization called the *bound sketch* optimization that was developed for the CBS estimator also to the optimistic estimators. Due to space limitations, we omit these applications in this version of our paper.

Table 2: Dataset descriptions.

Dataset	Domain	|V|	|E|	|E. Labels|
IMDb	Movies	27M	65M	127
YAGO	Knowledge Graph	13M	16M	91
DBLP	Citations	23M	56M	27
WatDiv	Products	1M	11M	86
Hetionet	Social Networks	45K	2M	24
Epinions	Consumer Reviews	76K	509K	50

6. EVALUATION

Recall that for pessimistic estimators, one should always pick the shortest path on CEG_M, but for optimistic estimators there is no obvious choice. We next present extensive experiments to evaluate the accuracies of the space of optimistic estimators we described on a large suite of datasets and workloads, so we can provide an advice based on an empirical justification. We note that the evaluation presented here is based on subgraph queries, which are equivalent to join queries over binary relations. Recall that this is because prior optimistic estimators were developed in this context. As a result, we note that the advice we give applies to this context. Using CEG-based optimistic estimators for queries over arbitrary relations is not in the scope of our paper.

Our original publication [9] contains many more experiments than we present here, including: (i) comparison of optimistic and pessimistic estimators (optimistic ones are much more accurate); (ii) impacts on the plan quality of using the optimistic estimators we advise on the RDF-3X system; (iii) scalability of Markov Tables and CEGs as the size of the patterns increase; and (iv) detailed demonstration of applying the bound sketch optimization of pessimistic estimator to optimistic estimators. Our code, datasets, and queries are publicly available [7].

6.1 Setup, Datasets and Workloads

For all of our experiments, we use a single machine with two Intel E5-2670 at 2.6GHz CPUs, each with 8 physical and 16 logical cores, and 512 GB of RAM. We used a total of 6 real-world datasets, shown in Table 2, and 6 workloads on these datasets. Our dataset and workload combinations are as follows.

IMDb and JOB [20]: The IMDb relational database, together with a workload called JOB (for join order benchmark), has been used for cardinality estimation studies in prior work [6, 20]. We created property graph versions of the this database and workload. The details of this con-

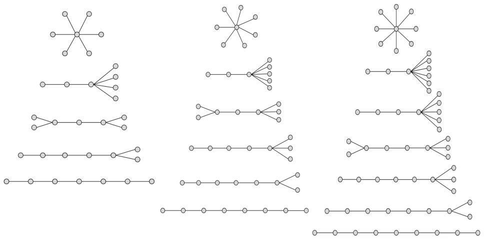


Figure 7: Query templates for the `Acyclic` workload. Edge directions are omitted in the figure.

struction can be found in our original publication [9]. Briefly, the construction is based on converting *entity tables*, representing actors, movies, and companies as *entity nodes*, and *relationship tables* representing many-to-many relationships between the entities as edges. In addition, adding *foreign key edges* between from an entity node u to v if there is a foreign key in the table u comes from to the table that v comes from. The final workload contained 369 queries.

WatDiv [35] and `WatDiv-Acyclic` Workload: WatDiv [4] is a synthetic knowledge graph that has its own workload in SPARQL format with 12400 original queries. We converted these queries into equivalent subgraph queries by removing their vertex predicates and we then removed queries with at most 3 edges. After removing duplicates among the remaining queries there were 75 different queries left, 9 of which is cyclic and the other 64 acyclic. Because 9 queries is very small for a workload, we use only the 64 acyclic queries call this the `WatDiv-Acyclic` workload.

YAGO 1 [40] and `G-CARE-Acyclic` and `G-CARE-Cyclic` Workloads: G-CARE [28] is a recent cardinality estimation benchmark for subgraph queries. From this benchmark we took the YAGO knowledge graph dataset and the acyclic and cyclic query workloads for that dataset. The `acyclic` workload contains 382 queries generated from query templates with 3-, 6-, 9-, and 12-edge star and path queries, as well as randomly generated trees. We will refer to this workload as `G-CARE-Acyclic`. The cyclic query workload, `G-CARE-Cyclic`, contains 240 queries generated from templates with 6-, and 9-edge cycle, 6-edge clique, 6-edge flower, and 6- and 9-edge petal queries.

DBLP [11], WatDiv [35], Epinions [12], and Hetionet [15] and `Acyclic` and `Cyclic` Workloads: We used three other datasets: (i) Hetionet: a biological network; (ii) DBLP: a real knowledge graph; and (iii) Epinions: a real-world social network graph. Epinions is a dataset that by default does not have any edge labels. We added a random set of 50 edge labels to Epinions. For these datasets we created one acyclic and one cyclic query workload, which we refer to as `Acyclic` and `Cyclic`. The `Acyclic` workload contains queries generated from 6-, 7-, or 8-edge templates, shown in Figure 7. Then, we generated 20 non-empty instances of each template by putting one edge label uniformly at random on each edge, which yielded 360 queries in total. The `Cyclic` workload contains queries generated from templates used in reference [24]. We generated instances of these queries by randomly matching each edge of the query template one at a time in the datasets. Because the WatDiv's original queries contained only acyclic queries, we used the `Cyclic` workload also on WatDiv. We generated 70 queries for DBLP, 212 queries for Hetionet, 129 queries for WatDiv, and 394 queries for Epinions.

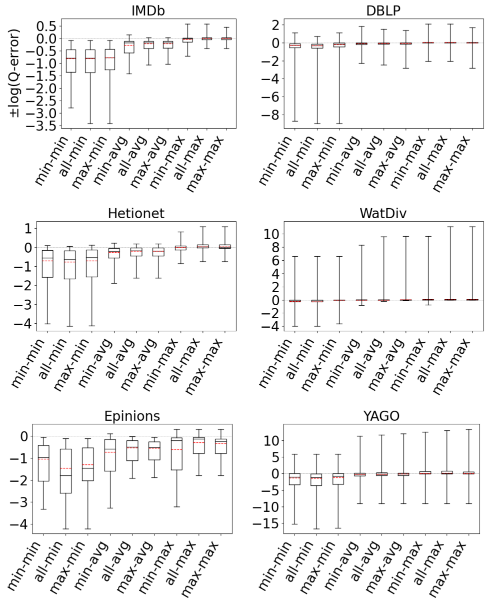


Figure 8: Evaluation of the optimistic estimators on CEG_O on acyclic queries. Estimators are labeled "P-A": P is the path length (one of max-hop, min-hop, or all-hops) and A the aggregator (one of max-aggr, min-aggr, or avg-aggr).

6.2 Space of Optimistic Estimators

We begin by comparing our 9 optimistic estimators on CEG_O, we defined. In order to set up an experiment in which we could test all of the 9 possible optimistic estimators, we used a Markov table with h=3. A Markov table with only 2-size joins can not test different estimators based on different path-length choices or any cyclic query.

To compare the accuracies of different estimators, for each query Q in our workloads we make an estimate using each estimator and compute its q-error. If the true cardinality of Q is c and the estimate is e, then the q-error is $\max\{\frac{c}{e}, \frac{e}{c}\} \geq 1$. For each workload, this gives us a distribution of q-errors, which we compare as follows. First, we take the logs of the q-errors so they are now ≥ 0. If a q-error was an underestimate, we put a negative sign to it. This allows us to order the estimates from the least accurate underestimation to the least accurate overestimation. We then generate a box plot where the box represents the 25th, median, and 75th percentile cut-off marks. The red dashed lines in figures are the means excluding the top 10% of the distribution (ignoring under/over estimations).

6.2.1 *Acyclic Queries and Cyclic Queries With Only Triangles*

Our first question is: Which of the 9 possible optimistic estimators leads to most accurate estimates on acyclic queries and cyclic queries that contain $\leq$3 edges on CEG_O? We compare our 9 estimators on CEG_O for each acyclic query workload in our setup (for IMDb, WatDiv, and YAGO, using JOB, `WatDiv-Acyclic`, and `G-CARE-Acyclic` workloads). We then compare our 9 estimators on each cyclic query workload, but only using the queries that only contain triangles as cycles. We omit `GCARE-Cyclic` because, except for one query every query in `GCARE-Cyclic` contained cycles with

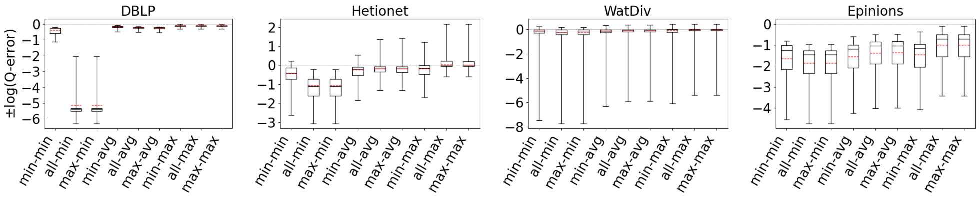


Figure 9: Evaluation of the space of optimistic estimators on CEG_O on `Cyclic` workload on queries with only triangles.

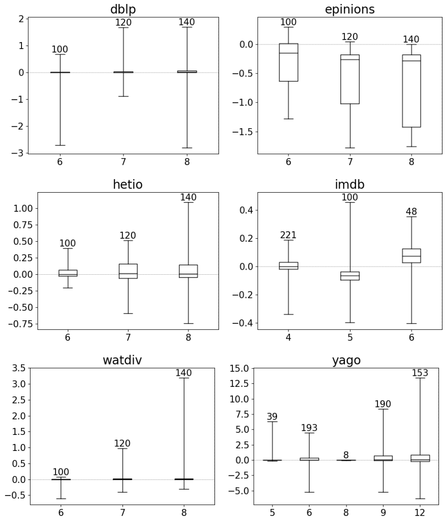


Figure 10: Accuracy of `max-max` for different query sizes (x-axis). The number on top of boxplots is the number of queries in the boxplot.

more than 3 edges.

Our results are shown in Figure 8. We make several observations. First, regardless of the path-length choice, the `max` aggregator (the last 3 box plots in the figures) makes significantly more accurate estimates (note that the y-axis on the plots are in log scale) than `avg`, which in turn is more accurate than `min`. This is true across all acyclic experiments and all datasets. For example, on IMDb and `JOB` workload, the `all-hops-min`, `all-hops-avg`, and `all-hops-max` estimators have log of mean q-errors of 6.5 (underestimation), 1.7 (underestimation), and 1.02 (understimation), respectively. *Therefore on acyclic queries, when there are multiple formulas that can be used to make an estimate, using the pessimistic ones is an effective technique to combat the well known underestimation problem for estimators that make uniformity and independence assumptions [18].* This can give up to three orders of magnitude lower mean q-errors than, e.g., the `min-hop-min` estimator from prior work.

We next analyze the effects of path-length choices. Observe that across all experiments, if we ignore the outliers and focus on the 25-75 percentile boxes, `max-hop` and `all-hops` do at least as well as `min-hop`. Further observe that on IMDb, Hetionet, and on the `Acyclic` workload on Epinions, `max-hop` and `all-hops` lead to significantly more accurate estimates. Finally, the performance of `max-hop` and `all-hops` are comparable across our experiments. We verified that this is because `all-hops` picks one of the `max-hop` paths in majority of the queries in our workloads. Therefore we observe that the advantage of long-hop paths that condition on 2-size joins when making uniformity assumptions is generally stronger than its disadvantage of making more independence assumptions. Since `max-hop` enumerates strictly fewer paths than `all-hops` to make an estimate, we conclude that on acyclic queries, systems implementing the optimistic estimators can prefer the `max-hop-max` estimator.

Figure 9 shows the accuracies of the 9 estimators on cyclic query workloads with only triangles. Our observations are similar to those for acyclic queries, and we find that the `max` aggregator yields more accurate estimates than other aggregators, irrespective of the path length. When using the max aggregator, we also observe that `max-hop` performs at least as well as `min-hop`. Therefore, as we observed for acyclic queries, *we find `max-hop-max` estimator to be an effective way to make accurate estimations for cyclic queries with only triangles.*

6.2.2 Cyclic Queries With Cycles of Size > 3

Recall our observation that on queries that contain large cycles, existing optimistic estimators estimate paths instead of cycles, which could yield highly inaccurate overestimates as real-world graphs contain many more paths than cycles. Therefore, unlike the case for acyclic queries and queries with small cycles, we now expect that estimates based on CEG_O will be pessimistic. To verify this hypothesis, our next experiments compare the performance of optimistic estimators for each dataset-cyclic query workload combination, but only using queries that contain cycles of size > 3.

Figure 11 shows our results. As we expected, we now see that the optimistic estimators yield overestimates for the majority of the queries. This can be seen by observing that except for a few exceptions the 25-75 percentile boxes of the boxplots are above 0. In addition, estimators using the `min` aggregator perform generally better, sometimes with several orders of magnitude difference, which can be seen by observing the mean accuracy lines of the boxplots. For example, on YAGO, the mean accuracy lines of `max-hop-min` and `max-hop-max` are, respectively, 0.20 and 3.61 on logarithmic scale, which correspond to absolute q-errors of 1.58 (overestimation) and 4043.86 (overestimation). We also observe that there is less sensitivity to the path-length choice when using the `min` aggregator, and any of the path-length choices perform reasonably well.

As we discuss in Section 8, we hope CEGs can be the foundation for proposing other estimators for future work. As a demonstration of the flexibility of our CEG-framework to design new CEG-based estimators, in the longer version of our paper [8], we present one possible approach (though others can be proposed) to remedy the overestimates of the estimators on CEG_O for queries with large cycle queries. Specifically, we describe a new CEG, that modifies CEG_O with new edge weights that incorporate a cycle closing effect and show that estimates on this new CEG can be more accurate than those on CEG_O.

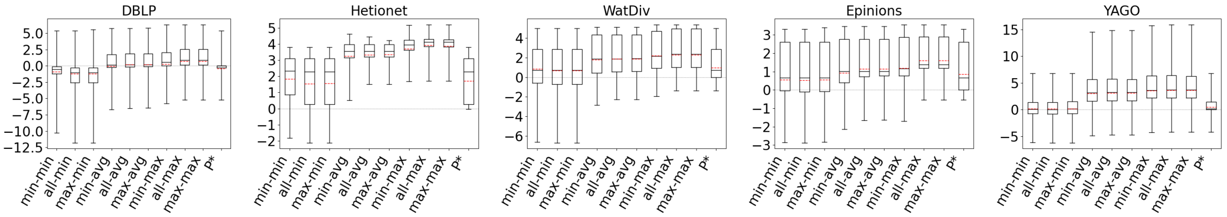


Figure 11: Evaluation of optimistic estimators on CEG_O on cyclic queries with cycles of 4 or more edges.

7. RELATED WORK

There is decades of extensive research on cardinality estimation techniques that can be used for estimating sizes of join queries. This literature has proposed several different classes of estimators, including other summary-based ones, as well as sampling-based ones and novel machine learning-based ones. Sampling-based estimators [10, 14, 17, 19, 21, 33, 38] sample input records from base tables and evaluate queries on these samples to make estimates. Sampling-based estimators are fundamentally different than summary-based ones as by increasing the sizes of the samples they can be arbitrarily accurate, while summary-based ones can be more accurate by keeping more statistics. Another class of estimators on which there is active work is machine-learning-based ones that learn to make estimates from example queries or predicates, e.g., example range predicates on columns [26, 36, 37]. Studying how all these estimators compare is beyond the scope of this work. In the following, we cover other summary-based estimators primarily focusing on graph-based database management systems.

Many relational systems use summary-based estimators. Example summaries include the cardinalities of relations, the number of distinct values in columns, or histograms [3, 25, 30], wavelets [23], or probabilistic and statistical models [13, 32] that capture the distribution of values in columns. These statistics are used to estimate the selectivities of each join predicate, which are put together using several approaches, such as independence assumptions. In contrast, the estimators we studied store degree statistics about base relations and small-size joins (note that cardinalities are a form of degree statistics, e.g., $|R_i|= deg(\emptyset, \mathcal{R}_i)$).

Characteristic Sets (CS) [27] is a summary-based estimator primarily designed to estimate the cardinalities of stars in an RDF graph. It uses the so-called *characteristic set* of each vertex v in an RDF graph, which is the set of distinct outgoing labels v has. CS keeps statistics about the vertices with the same characteristic set. Then, using these statistics, CS makes estimates for the sizes of star queries. For a non-star query Q, Q is decomposed into multiple stars $s_1, ..., s_k$ in a greedy manner, by removing the largest stars first, and the estimates for each s_i is multiplied. However, unlike the optimistic estimators we considered, this decomposition procedure does not lead to multiple possible decompositions.

Several works have proposed summary-based estimators that compute a sketch of an input graph. SumRDF [31] builds a summary graph S of an RDF graph and adopts a holistic approach to make an estimate. Given the summary S, SumRDF considers all possible RDF graphs G that could have the same summary S. In the context of estimating the selectivities of path expressions, XSeed [41] and XSketch [29] build a sketch S of the input XML Document. The sketch of the graph effectively collapses multiple nodes and edges into supernodes and edges with statistics on them. Then a query Q is matched on S and an estimate is made using the metadata. These techniques do not decompose a query into smaller sub-queries, so the question of which decomposition to use does not arise for these estimators.

Similarly, several work use data structures that are adaptations of histograms from relational systems to store selectivities of paths or trees in XML documents. Examples include, *positional histograms* [39] and *Bloom histogram* [34]. These techniques do not consecutively make estimates for larger paths and have not been adopted to general subgraph queries.

8. LIMITATIONS AND FUTURE WORK

Aside from capturing existing optimistic and pessimistic estimators, we believe the CEG framework can be the foundation to develop novel summary-based estimators. In addition to the two CEGs we considered here, CEG_O and CEG_M, other CEGs using different statistics can be defined and paired with different techniques to pick paths. To demonstrate an example, in the longer version of our paper [8] we describe another CEG we call CEG_{OCR} as a possible approach to remedy the overestimation problem of optimistic estimator on CEG_O for queries with large cycles. CEG_{OCR} uses new edge weights that capture the closing of large cycles between sub-queries.

Perhaps the most important research question our work leaves unanswered is: which CEG should be used in practice? This is a very broad question but it is easy to define many other CEGs and our understanding of which ones would yield better accuracy is limited. For example, one can use variances or entropies of the distributions of small-size joins as edge weights, possibly along with degree statistics, and pick the lowest entropy paths. An important research direction is to systematically study a class of CEG instances that use different combination of statistics as edge weights, as well as techniques for picking paths, to design more accurate CEG-based estimators. In addition, in this work, we focused only on join-only queries and ignore other non-join predicates. An additional future work direction is a principled mechanisms to integrate filters on the queries that can be estimated using CEG.

9. REFERENCES

[1] M. Abo Khamis, H. Q. Ngo, and D. Suciu. Computing Join Queries with Functional Dependencies. In *PODS*, 2016.

[2] A. Aboulnaga, A. R. Alameldeen, and J. F. Naughton. Estimating the Selectivity of XML Path Expressions for Internet Scale Applications. In *VLDB*, 2001.

[3] A. Aboulnaga and S. Chaudhuri. Self-Tuning Histograms: Building Histograms Without Looking at Data. In *SIGMOD*, 1999.

[4] G. Aluç, O. Hartig, M. T. Özsu, and K. Daudjee. Diversified Stress Testing of RDF Data Management Systems. In *ISWC*, 2014.

[5] A. Atserias, M. Grohe, and D. Marx. Size Bounds and Query Plans for Relational Joins. *SICOMP*, 42(4), 2013.

[6] W. Cai, M. Balazinska, and D. Suciu. Pessimistic Cardinality Estimation: Tighter Upper Bounds for Intermediate Join Cardinalities. In *SIGMOD*, 2019.

[7] CEG Evaluation Source Code. https://github.com/cetechreport/CEExperiments, 2022.

[8] J. Chen, Y. Huang, W. Mushi, S. Semih, and S. Ken. Accurate Summary-based Cardinality Estimation Through the Lens of Cardinality Estimation Graphs https://cs.uwaterloo.ca/~ssalihog/papers/ceg-long.pdf. Technical report, January 2022.

[9] J. Chen, Y. Huang, M. Wang, S. Salihoglu, and K. Salem. Accurate Summary-Based Cardinality Estimation through the Lens of Cardinality Estimation Graphs. *PVLDB*, 15(8), 2022.

[10] Y. Chen and K. Yi. Random Sampling and Size Estimation Over Cyclic Joins. In *ICDT*, 2020.

[11] DBLP 2012-11-28 Dump. https://dblp.org/, 2012.

[12] Epinions. https://snap.stanford.edu/data/soc-Epinions1.html, 2003.

[13] L. Getoor, B. Taskar, and D. Koller. Selectivity Estimation Using Probabilistic Models. In *SIGMOD*, 2001.

[14] Haas, Peter J. and Naughton, Jeffrey F. and Seshadri, S. and Swami, Arun N. Selectivity and Cost Estimation for Joins Based on Random Sampling. *JCSS*, 52(3), 1996.

[15] Hetionet v1.0. https://het.io/, 2015.

[16] M. Joglekar and C. Ré. It's All a Matter of Degree - Using Degree Information to Optimize Multiway Joins. *TOCS*, 62(4), 2018.

[17] K. Kim, H. Kim, G. Fletcher, and W.-S. Han. Combining sampling and synopses with worst-case optimal runtime and quality guarantees for graph pattern cardinality estimation. In *SIGMOD*, 2021.

[18] V. Leis, A. Gubichev, A. Mirchev, P. Boncz, A. Kemper, and T. Neumann. How Good Are Query Optimizers, Really? *PVLDB*, 9(3), 2015.

[19] V. Leis, B. Radke, A. Gubichev, A. Kemper, and T. Neumann. Cardinality Estimation Done Right: Index-Based Join Sampling. In *CIDR*, 2017.

[20] V. Leis, B. Radke, A. Gubichev, A. Mirchev, P. Boncz, A. Kemper, and T. Neumann. Query Optimization Through the Looking Glass, and What We Found Running the Join Order Benchmark. *VLDBJ*, 27(5), 2018.

[21] F. Li, B. Wu, K. Yi, and Z. Zhao. Wander Join: Online Aggregation via Random Walks. In *SIGMOD*, 2016.

[22] A. Maduko, K. Anyanwu, A. Sheth, and P. Schliekelman. Graph Summaries for Subgraph Frequency Estimation. In *ESWC*, 2008.

[23] Y. Matias, J. S. Vitter, and M. Wang. Wavelet-Based Histograms for Selectivity Estimation. In *SIGMOD*, 1998.

[24] A. Mhedhbi and S. Salihoglu. Optimizing Subgraph Queries by Combining Binary and Worst-Case Optimal Joins. *PVLDB*, 12(11), 2019.

[25] M. Muralikrishna and D. J. DeWitt. Equi-Depth Histograms for Estimating Selectivity Factors for Multi-Dimensional Queries. In *SIGMOD*, 1988.

[26] P. Negi, R. Marcus, H. Mao, N. Tatbul, T. Kraska, and M. Alizadeh. Cost-guided cardinality estimation: Focus where it matters. In *ICDEW*, 2020.

[27] T. Neumann and G. Moerkotte. Characteristic Sets: Accurate Cardinality Estimation for RDF Queries with Multiple Joins. In *ICDE*, 2011.

[28] Y. Park, S. Ko, S. S. Bhowmick, K. Kim, K. Hong, and W.-S. Han. G-CARE: A Framework for Performance Benchmarking of Cardinality Estimation Techniques for Subgraph Matching. In *SIGMOD*, 2020.

[29] N. Polyzotis and M. Garofalakis. Statistical Synopses for Graph-Structured XML Databases. In *SIGMOD*, 2002.

[30] V. Poosala and Y. E. Ioannidis. Selectivity Estimation Without the Attribute Value Independence Assumption. In *VLDB*, 1997.

[31] G. Stefanoni, B. Motik, and E. V. Kostylev. Estimating the Cardinality of Conjunctive Queries over RDF Data Using Graph Summarisation. In *WWW*, 2018.

[32] W. Sun, Y. Ling, N. Rishe, and Y. Deng. An Instant and Accurate Size Estimation Method for Joins and Selections in a Retrieval-Intensive Environment. In *SIGMOD*, 1993.

[33] D. Vengerov, A. C. Menck, M. Zait, and S. P. Chakkappen. Join Size Estimation Subject to Filter Conditions. *PVLDB*, 8(12), 2015.

[34] W. Wang, H. Jiang, H. Lu, and J. X. Yu. Bloom Histogram: Path Selectivity Estimation for XML Data with Updates. In *VLDB*, 2004.

[35] WatDiv v.0.6. https://dsg.uwaterloo.ca/watdiv/, 2014.

[36] L. Woltmann, C. Hartmann, M. Thiele, D. Habich, and W. Lehner. Cardinality estimation with local deep learning models. In *aiDM*, 2019.

[37] L. Woltmann, D. Olwig, C. Hartmann, D. Habich, and W. Lehner. PostCENN: PostgreSQL with Machine Learning Models for Cardinality Estimation. *PVLDB*, 14(12), 2021.

[38] W. Wu, J. F. Naughton, and H. Singh. Sampling-Based Query Re-Optimization. In *SIGMOD*, 2016.

[39] Y. Wu, J. M. Patel, and H. V. Jagadish. Estimating Answer Sizes for XML Queries. In *EDBT*, 2002.

[40] YAGO 1. https://yago-knowledge.org/downloads/yago-1, 2008.

[41] N. Zhang, M. T. Ozsu, A. Aboulnaga, and I. F. Ilyas. XSEED: Accurate and Fast Cardinality Estimation for XPath Queries. In *ICDE*, 2006.