Fractional XSKETCH Synopses for XML Databases

Natasha Drukh[1], Neoklis Polyzotis[2], Minos Garofalakis[3], and Yossi Matias[1]

[1] School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel,
kreimern@post.tau.ac.il, matias@cs.tau.ac.il
[2] Dept. of Computer Science, Univ. of California - Santa Cruz, Santa Cruz, CA 95064, USA,
alkis@cs.ucsc.edu
[3] Bell Labs, Lucent Technologies, Murray Hill, NJ 07974, USA,
minos@research.bell-labs.com

Abstract. A key step in the optimization of declarative queries over XML data is estimating the selectivity of path expressions, i.e., the number of elements reached by a specific navigation pattern through the XML data graph. Recent studies have introduced XSKETCH structural graph synopses as an effective, space-efficient tool for the compile-time estimation of complex path-expression selectivities over graph-structured, schema-less XML data. Briefly, XSKETCHes exploit localized graph stability and well-founded statistical assumptions to accurately approximate the path and branching distribution in the underlying XML data graph. Empirical results have demonstrated the effectiveness of XSKETCH summaries over real-life and synthetic data sets, and for a variety of path-expression workloads.
In this paper, we introduce fractional XSKETCHes (fXSKETCHES) a simple, yet intuitive and very effective generalization of the basic XSKETCH summarization mechanism. In a nutshell, our fXSKETCH synopsis extends the conventional notion of binary stability (employed in XSKETCHes) with that of *fractional stability*, essentially recording more detailed path/branching distribution information on individual synopsis edges. As we demonstrate, this natural extension results in several key benefits over conventional XSKETCHes, including (a) a simplified estimation framework, (b) reduced run-time complexity for the synopsis-construction algorithm, and (c) lifting the need for critical uniformity assumptions during estimation (thus resulting in more accurate estimates). Results from an extensive experimental study show that our fXSKETCH synopses yield significantly better selectivity estimates than conventional XSKETCHes, especially in the context of complex path expressions with branching predicates.

1 Introduction

XML has rapidly evolved from a mark-up language to a de-facto standard for data exchange and integration over the web. A testament to this is the increasing volume of published XML data, together with the concurrent development of XML query processors that will allow users to tap into the vast amount of XML data available on the Internet. The successful deployment of such query processors depends crucially on the existence of high-level declarative query languages. There exist numerous proposals that cover a wide range of paradigms, but a common characteristic among all XML-language proposals is the use of *path expressions* as the basic method to access and retrieve specific

Z. Bellahsène et al. (Eds.): XSym 2004, LNCS 3186, pp. 189–203, 2004.

elements from the XML database. A path expression essentially defines a complex navigational path, which can be predicated on the existence of sibling paths or on constraints on the values of visited elements. As a concrete example, in a bibliography database, the path expression `//author[book]/paper/sigmod/title` (which adheres to the syntax of the standard XPath language [1]) selects the set of all `title` data elements discovered by the label path `//author/paper/sigmod/title`, but only for `author` elements that have *at least one* `book` child (a condition specified by the `author[book]` branch).

Similar to relational optimization, optimizing XML queries with complex path expressions depends crucially on the existence of concise summaries that can provide effective compile-time estimates for the selectivity of these expressions over the underlying (large) graph-structured XML database. This problem has recently attracted the attention of the database research community, and several techniques [2,3,4,5,6,7, 8] have been proposed targeting different aspects of the problem. XSKETCH structural graph synopses [5,6] have recently been introduced as an effective data-reduction tool that enables accurate selectivity estimates for branching path expressions. In a nutshell, XSKETCH synopses exploit localized graph stability and well-founded statistical assumptions to accurately approximate the path and branching distribution in the underlying XML data graph; furthermore, XSKETCHes can be augmented with summary information on data-value distributions to handle path expressions with value predicates [6]. Compared to previously proposed techniques, the XSKETCH synopsis mechanism targets the most general version of the estimation problem: XPath expressions with branching and value predicates, over graph-structured, schema-less XML databases. Experimental results with a variety of query workloads on different data sets have demonstrated the effectiveness of XSKETCHes as concise summaries of XML data.

In this paper, we introduce *fractional* XSKETCHes (fXSKETCHES) a simple, yet intuitive and very effective generalization of the basic XSKETCH synopses based on the concept of *fractional edge stabilities*. Briefly, instead of simply recording whether a synopsis edge is stable or not (i.e., the conventional "binary" notion of stability employed in the XSKETCH model), our fXSKETCH synopses record the *degree of stability* for each edge as a *fraction* between 0 ("no-connection") and 1 ("fully stable"). As we demonstrate, this natural generalization has a direct positive impact on the underlying estimation framework. First, it simplifies the expressions for query-selectivity estimates, thus allowing for faster estimation. Second, and perhaps most importantly, it lifts the need for certain critical uniformity assumptions during basic XSKETCH estimation, thus resulting in significantly more robust and accurate estimates. Furthermore, the removal of such uniformity assumptions also reduces the search space (and, therefore, the time complexity) of the synopsis-construction algorithm, since it effectively obviates the need for specialized synopsis-refinement operations to address regions of non-uniformity. These observations are backed up by an extensive experimental study which evaluates the performance of our generalized fXSKETCH synopses on a variety of XML data sets and query workloads. Our results clearly indicate that fXSKETCHES yield significant improvements in accuracy when compared to original XSKETCH summaries. These improvements are more apparent in the case of complex path expressions with branching predicates, where the uniformity assumptions of the original XSKETCH model can introduce large errors;

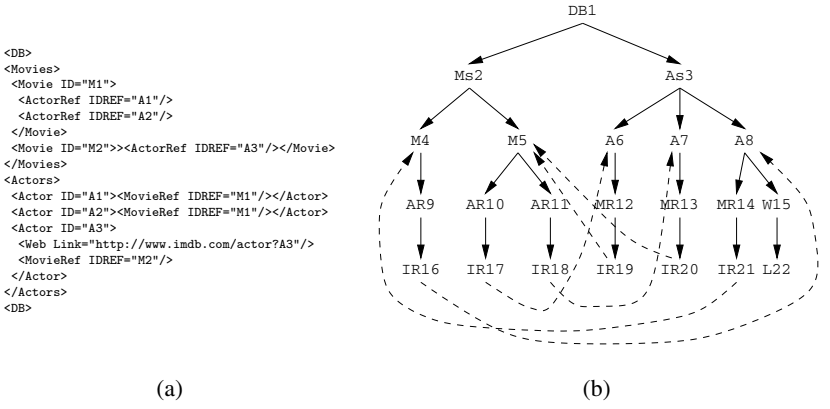


Fig. 1. Example XML document (a) and XML data graph (b).

fractional stabilities, on the other hand, lift the need for such assumptions thus resulting in significantly better fXSKETCH-based selectivity estimates.

The remainder of this paper is organized as follows. Section 2 covers some preliminary material on XML and path expressions, while Section 3 provides a short overview of the original XSKETCH model [5,6]. Our generalized fXSKETCH synopsis model is described in Section 4, where we discuss the definition of fractional stabilities and their implications on the estimation framework and the synopsis-construction process. Section 5 presents the results of our experimental study, while Section 6 gives some concluding remarks and our plans for future work.

2 Preliminaries

XML Data Model. Following previous work on XML and semistructured data [9,10], we model an XML database as a large, directed, node-labeled *data graph* $G = (V_G, E_G)$. Each node in V_G corresponds to an XML element in the database and is characterized by a *unique object identifier (oid)* and a *label* (assigned from some alphabet of string literals) that captures the semantics of the element. (We use $\texttt{label}(v)$ to denote the label of node $v \in V_G$.) Edges in E_G are used to capture both the element-subelement relationships (i.e., element nesting) and the explicit element references (i.e., `id`/`idref` attributes or XLink constructs [11,12,9,13]). Note that non-tree edges, such as those implemented through `id`/`idref` constructs, are an essential component and a "first-class citizen" of XML data that can be directly queried in complex path expressions, such as those allowed by the XQuery standard specification [14]. We, therefore, focus on the most general case of XML data *graphs* (rather than just trees) in what follows.

Example 1. Figures 1(a,b) show an example XML document and its corresponding data graph. The document is modeled after the Internet Movie Database (IMDB) XML data set (`www.imdb.com`), showing two movies and three actors. The graph node corresponding to a data element is named with an abbreviation of the element's label and a unique id number. Note that we use dashed lines to show graph edges that correspond to `id-idref` relationships.

XPath Expressions. Abstractly, an XML path expression $\bar{\mathtt{l}}$ (e.g., in XQuery [14]) defines a navigational path over the XML data graph, specifying conditions on the labels and (possibly) the value(s) of data elements. Following the XPath standard [1], a *simple path expression* is of the form $\mathtt{l}_1/\mathtt{l}_2/\ldots/\mathtt{l}_n$, where the $\mathtt{l}_i$'s are document labels. The result of the path expression includes all elements u_n for which there exists a document path $u_1/u_2/\ldots/u_n$ with $\mathtt{label}(u_i) = \mathtt{l}_i$. A *branching path expression* has the form $\bar{\mathtt{l}} = \mathtt{l}_1[\bar{\mathtt{l}}^1]/\ldots/\mathtt{l}_n[\bar{\mathtt{l}}^n]$, where the l_i's are labels and each $\bar{\mathtt{l}}^i$ is a (possibly empty) label path specifying a branching path predicate at location i. Thus, a branching path expression is formed from a simple path expression $\mathtt{l}_1/\mathtt{l}_2/\ldots/\mathtt{l}_n$ by attaching (one or more) branching predicates $[\bar{\mathtt{l}}^i]$ at specific nodes in the path. Each $[\bar{\mathtt{l}}^i]$ clause represents an existential condition, requiring that there exists *at least one* $\bar{\mathtt{l}}^i$ label branch attached at point i of the expression. For example, consider the document graph of Figure 1 and the simple path expression `Actor/MovieRef/IDREF/Movie` that retrieves elements with ids 4 and 5; if we add a `[Link]` branch on `Actor`, then the new path expression `Actor[Link]/MovieRef/IDREF/Movie` only retrieves element 4. Note that if all branch predicates are empty, a branching path expression degenerates to a simple path expression $\mathtt{l}_1/\cdots/\mathtt{l}_n$.

It is possible to extend branching path expressions with predicates on the *values* of traversed elements. As an example, the XPath expression `Actor[/MovieRef/IDREF/Movie/Title=''Snatch'']` retrieves all actors that have starred in a movie with title "Snatch". In the interest of space, we ignore element values when we discuss the specifics of our fXSKETCH synopses, focusing primarily on the label path and branching structure of the underlying database – the necessary extensions to handle values and value predicates follow along similar lines as the analogous extensions for basic XSKETCHes [6].

3 A Review of XSKETCH Synopses

Synopsis Model. The XSKETCH synopsis mechanism [5,6] relies on a generic graph-summary model that captures the basic path structure of the input XML tree. Formally, given a data tree $G = (V_G, E_G)$, a graph synopsis $\mathcal{S}(G) = (V_\mathcal{S}, E_\mathcal{S})$ is a directed node-labeled graph, where (1) each node $v \in V_\mathcal{S}$ corresponds to a subset of element (or, attribute) nodes in V_G (termed the *extent* of v – $\mathtt{extent}(v)$) that have the *same label*, and (2) an edge in $(u, v) \in E_G$ is represented in $E_\mathcal{S}$ as an edge between the synopsis nodes whose extents contain the two endpoints u and v. Each synopsis node u stores the (common) label $\mathtt{label}(u)$ of all elements in its extent, and an element-count field $|u| = |\mathtt{extent}(u)|$ (we use u and $\mathtt{extent}(u)$ interchangeably in what follows). Figure 2(a) shows a graph synopsis for the document of Figure 1, where elements are partitioned according to their label (synopsis nodes are named with the first letter of their label in upper case).

XSKETCH synopses [5,6] are specific instantiations of the graph-synopsis model described above. In order to cover key properties of the path and branching distribution, the basic synopsis is augmented with edge labels that capture localized *backward- and forward-stability* [15] conditions across synopsis nodes. An edge $u \rightarrow v$ is **B**ackward (resp., **F**orward) stable, if all elements in the extent of v (resp., u) have at least one parent (resp., child) element in the extent of u (resp., v). As an example, Figure 2(b) shows the XSKETCH summary for the graph synopsis of Figure 2(a). Note that edge $\mathtt{A} \rightarrow \mathtt{MR}$ is both

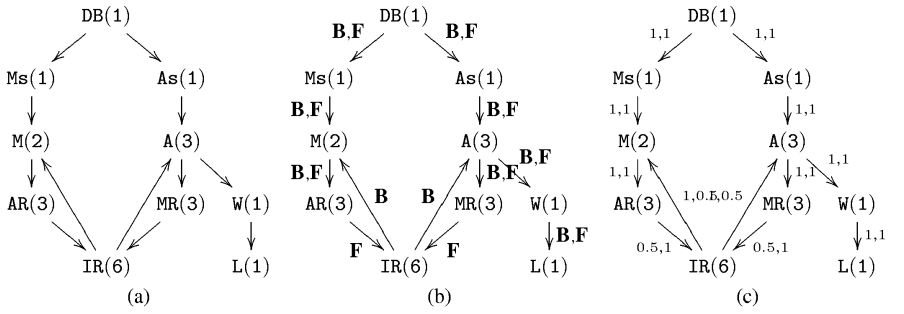


Fig. 2. (a) Graph Synopsis, (b) XSKETCH Synopsis, (c) fXSKETCH Synopsis.

backward and forward stable since all `MovieRefs` have an `Actor` parent, and all `Actors` have at least one `MovieRef` child. As a result, $|\texttt{MR}| = 3$ is an accurate selectivity estimate for path expression `A/MR`, while $|\texttt{A}| = 3$ is an accurate estimate for `A[/MR]`. As shown in [5,6], such *localized* stability information can be combined to effectively capture key properties of the *global* path structure, and provide accurate estimates on the selectivity of XPath expressions over the original XML document graph.

Estimation Framework. The XSKETCH framework uses the concept of *path embeddings* in order to estimate the selectivity of any branching path expression. In short, a synopsis path $u_1/\ldots/u_k$ is called an embedding of the simple path expression $\texttt{l}_1/\ldots/\texttt{l}_k$, if $\texttt{label}(u_i) = \texttt{l}_i$ for each $1 \leq i \leq k$. Similarly, a synopsis twig $\bar{u} = u_1[\bar{u}^1]/\ldots/u_k[\bar{u}^k]$, where $\bar{u}^i$ is a synopsis path, is called an embedding of the branching path expression $\bar{l} = \texttt{l}_1[\bar{\texttt{l}}^1]/\ldots/\texttt{l}_k[\bar{\texttt{l}}^k]$, if $u_1/\ldots/u_k$ is an embedding of $\texttt{l}_1/\ldots/\texttt{l}_k$ and $\bar{u}^i$ is an embedding of $\bar{\texttt{l}}^i$, for each $1 \leq i \leq k$. The selectivity of an XPath expression can be estimated as the sum of selectivities of its unique embeddings, and the problem is thus reduced to estimating the selectivity of a single embedding.

To estimate the selectivity of a single embedding, the XSKETCH estimation algorithms identify sub-components comprising only stable edges (for which accurate estimates can be given based on edge stability), and then apply statistical (uniformity and independence) assumptions at the "breaking" points of the stability chain(s). A detailed description of the estimation framework can be found in [5,6]; here, we use a simple example to illustrate the basic concepts. Consider the synopsis of Figure 2(b) and the path embedding `M/AR/IR[/A/MR]`. We express the selectivity of the embedding as $|\texttt{IR}| \cdot f(M/AR/IR[/A/MR])$, where the $f()$ term denotes the fraction of elements in `IR` that are reached by the embedding. Using the *Chain Rule* from probability theory, this fraction can be written as follows:

$$\begin{aligned} f(M/AR/IR[/A/MR]) &= f(M/AR) \cdot f(AR/IR \mid M/AR) \cdot f(IR[/A/MR] \mid M/AR/IR) \\ &= f(M/AR) \cdot f(AR/IR \mid M/AR) \cdot f(IR[/A] \mid M/AR/IR) \cdot \\ &\quad f(A[/MR] \mid M/AR/IR[/A]). \end{aligned}$$

We observe that, by virtue of **B**-stability, the term $f(M/AR)$ is equal to 1 (all elements in `IR` have a parent in `AR`); similarly, by virtue of **F**-stability, the term $f(A[/MR] \mid M/AR/IR[/A])$ is also equal to 1 (all elements in `A` have at least one child in `MR`).

The remaining two terms, however, cannot be computed based solely on available stability annotations. To compensate for this lack of path-distribution information, the XSKETCH estimation framework applies an *independence assumption*, that de-correlates the needed fractions from incoming or outgoing paths as follows: $f(AR/IR \mid M/AR) \approx f(AR/IR)$, $f(IR[/A] \mid M/AR/IR) \approx f(IR[/A])$. Thus, our selectivity expression is written as:

$$f(M/AR/IR[/A/MR]) = f(AR/IR) \cdot f(IR[/A]).$$

To approximate the required fractions, the XSKETCH estimation algorithm applies two *uniformity assumptions* based on the stored node counts. Since these assumptions are central in the theme of this paper, we define them formally below.

A1 [Backward-Edge Uniformity Assumption]. Given an XSKETCH node v, the incoming edges to v from all parent nodes u of v such that v is *not* **B**-stable with respect to u are *uniformly distributed* across all such parents in proportion to their element counts.

A2 [Forward-Edge Uniformity Assumption]. Given an XSKETCH node v, the outgoing edges from v to all children u of v such that v is *not* **F**-stable with respect to u are *uniformly distributed* across all such children in proportion to their element counts, and the total number of such edges is *at most* equal to the total of these element counts.

Returning to our example, Assumption **A1** provides the approximation $f(AR/IR) \approx |AR|/(|AR| + |MR|) = 3/6$, while Assumption **A2** yields $f(\exists\ IR/A) \approx |A|/\max\{|A| + |M|, |IR|\} = 3/6$. Thus, the estimate of the fraction is 0.25, and the overall path-expression selectivity can be approximated as $|IR| * 0.25 = 1.5$

Overall, the XSKETCH estimation framework uses stabilities in order to identify fully-stable subpath embeddings (for which accurate estimates can be given), and resorts to statistical assumptions to compensate for the lack of detailed path-distribution information in the synopsis. Clearly, the validity of these assumptions directly affects the accuracy of the resulting estimates. As we discuss next, the XSKETCH framework addresses this issue during the construction phase, where the goal is to find a "good" partitioning of elements into synopsis nodes such that the underlying estimation assumptions are valid [5,6].

XSKETCH**Construction.** At an abstract level, the XSKETCH construction problem can be defined as follows: Given a document graph G and a space budget S, build an XSKETCH synopsis of G that requires at most S storage of units, while minimizing the estimation error. Given the specifics of the XSKETCH model, this can be re-stated as follows: Compute a partitioning of data elements into synopsis nodes, such that the resulting XSKETCH (a) needs at most S units of storage, and (b) maximizes the validity of the estimation assumptions, thus minimizing error. Given that finding an optimal solution is typically an $\mathcal{NP}$-hard problem [5], the proposed XSKETCH construction algorithm (termed BUILDXSKETCH [5]) employs a heuristics-based, greedy search strategy in order to construct an effective, concise summary. In what follows, we provide a brief description of the key concepts behind the BUILDXSKETCH algorithm (more details can be found in [5]).

BUILDXSKETCH constructs an XSKETCH summary by incrementally refining a coarse synopsis, until it exhausts the available space budget. The starting summary assigns elements to synopsis nodes based solely on their label and, thus, represents a very coarse partitioning of the input document graph. At each step, this coarse partitioning is refined by applying one of three types of *refinement operations*, namely `b-stabilize`, `f-stabilize`, and `b-split`, on a specific synopsis node. Such refinement operations split the synopsis node (according to a specific criterion), resulting in a more refined partitioning. The split criterion is directly tied to the estimation assumptions that the refinement targets. For instance, the `b-stabilize` operation splits the node so that one of the two new nodes becomes **B**-stable with respect to a particular parent; in this manner, **B**-stability is now guaranteed along the new edge and the new summary is expected to have better estimates in the refined area. To select one of the possible refinements at each step, the BUILDXSKETCH algorithm employs a *marginal-gains* strategy: the refinement that yields the largest increase in accuracy per unit of additional required storage is chosen. Intuitively, this strategy leads to a summary which is more refined where correlations are stronger (and, thus, estimation assumptions are less valid), and less refined where the independence and uniformity assumptions provide good estimates of the true selectivities.

4 The fXSKETCH Model

As discussed in Section 3, the basic XSKETCH model employs the conventional, "binary" form of edge stabilities [15] in order to capture the key properties of the underlying path and branching structure. If stability is present (i.e., a value of 1), then there are strong guarantees on the connectivity between elements of edge-connected synopsis nodes; if, on the other hand, it is absent (i.e., a value of 0), then the XSKETCH estimation framework needs to apply independence and uniformity in order to approximate the true selectivity of the corresponding edge condition. In this section, we propose a simple, yet powerful generalization of the basic XSKETCH model based on a novel, more flexible notion of *fractional stabilities*. In a nutshell, instead of treating edge stability as a binary attribute, our new synopsis model uses a real number which reflects the *degree of stability* for each synopsis edge. In this manner, the synopsis stores distribution information at a finer level of detail, thus increasing the accuracy of the overall approximation.

Before describing our proposed synopsis mechanism in more detail, we present a simple example that illustrates the motivation behind fractional stabilities. Consider the sample document of Figure 3(a), where the numbers along edges denote the numbers of child elements. Figure 3(b) shows the coarsest XSKETCH synopsis, which groups together elements according to their tags (for simplicity, we omit **F**-stabilities from the figure). Note that the basic XSKETCH estimation framework has to apply a backward-edge uniformity assumption (**A1**) in order to estimate the number of `f` elements at the end of a path expression. Considering the skew in element counts, however, this assumption obviously introduces large errors in the estimate; as an example, the selectivity fraction of the embedding `B/F` is estimated as $f(B/F) = 1/(1+1+1+1) = 0.25$, while its true value is only 10^{-4}! In order to capture such skew, the XSKETCH construction algorithm would have to apply successive stabilization operations, in order to separate the `f` elements according to their incoming path. This, however, would lead to a finer

partitioning, thus inevitably increasing the storage requirements of the synopsis. In addition, stabilization operations mainly target independence assumptions and it is not clear if the greedy construction algorithm can actually discover their relation to an invalid uniformity assumption (in the sample document, for example, independence is in fact valid). Note that, although we focused our example on backward uniformity, a similar argument can be made for forward-edges as well (Assumption **A2**).

As a solution to this shortcoming of the original XSKETCH model, we propose to store more detailed information along the edges of the graph synopsis in the form of *fractional stabilities*. More formally, these are defined as follows.

Definition 1. *Let $\mathcal{S}(G) = (V_\mathcal{S}, E_\mathcal{S})$ be a graph synopsis and $(u, v) \in E_\mathcal{S}$ be a synopsis edge. The fractional* **B***-stability of (u, v), denoted as $\mathbf{B}_q(u, v)$, is the fraction of elements in v that have at least one parent in u. The fractional* **F***-stability of (u, v), denoted as $\mathbf{F}_q(u, v)$, is the fraction of elements in u that have at least one child in v.*

Clearly, our new model of fractional edge stabilities subsumes conventional, binary stabilities. More specifically, an edge (u, v) is **B**-stable (resp., **F**-stable) if and only if $\mathbf{B}_q(u, v) = 1$ (resp., $\mathbf{F}_q(u, v) = 1$), and unstable otherwise. Moreover, it is interesting to note that, by definition, fractional stabilities always provide *zero-error estimates* for path embeddings of length 2. For instance, the selectivity of an embedding u/v can be accurately estimated as $|v| \cdot \mathbf{B}_q(u, v)$, while for the length-2 branch $u[v]$, the selectivity is simply equal to $|u| \cdot \mathbf{F}_q(u, v)$. At this point, we can formally define our novel synopsis model of *fractional* XSKETCH*es (f*XSKETCHES*)* as follows.

Definition 2. *An f*XSKETCH *summary $\mathcal{S}(G) = (V_\mathcal{S}, E_\mathcal{S})$ of a document graph G is a graph synopsis of G that records the fractional stabilities $\mathbf{B}_q(u, v)$ and $\mathbf{F}_q(u, v)$ for each edge $(u, v) \in E_\mathcal{S}$.*

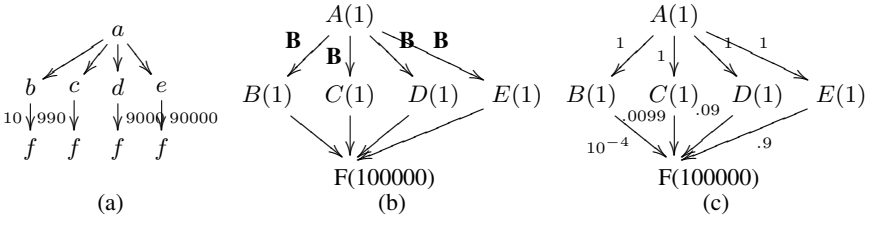


Fig. 3. (a) XML Document, (b) XSKETCH, (c) fXSKETCH.

Figure 3(c) depicts an example fXSKETCH for the document of Figure 3(a) (the edges are annotated with fractional **B**-stabilities only, since all fractional **F**-stabilities are equal to 1). Obviously, assuming a fixed synopsis-graph structure, an fXSKETCH has increased storage requirements when compared against the corresponding XSKETCH: instead of storing only two bits per edge (to denote absence or presence of **B/F**-stability), each fXSKETCH edge needs to record two real numbers (i.e., the new fractional stabilities). This finer level of detail, however, allows a concise fXSKETCH summary to capture

richer information in limited space, without resorting to a finer partitioning of elements to synopsis nodes. Returning to the example of Figure 3, we observe that the fXSKETCH contains accurate information for all paths of length up to 2, while maintaining the coarsest partitioning of elements to synopsis nodes; the XSKETCH, on the other hand, will exhibit high estimation errors due to an inaccurate uniformity assumption and, as mentioned earlier, can capture skew only by resorting to a much finer partitioning of elements. This intuitive advantage of fractional stabilities is corroborated by our experimental findings, where our fXSKETCH synopses consistently return significantly more accurate selectivity estimates for the same synopsis space budget (Section 5).

We now discuss how our new fractional stabilities are integrated in the general XSKETCH framework for path-expression selectivity estimation. Once again, the key observation here is that fractional stabilities essentially provide *zero-error estimates* for the selectivities of single-edge path embeddings. More formally, if (u, v) is any edge in the fXSKETCH synopsis, then the selectivity fractions $f(u/v)$ and $f(u[v])$ are simply defined as $f(u/v) = \mathbf{B}_q(u, v)$ and $f(u[v]) = \mathbf{F}_q(u, v)$. Thus, instead of applying uniformity assumptions to approximate such terms (possibly incurring large estimation errors), our fXSKETCH estimation algorithm can retrieve the accurate information directly from the stored fractional stabilities. As an example, consider again the embedding `M/AR/IR[A/MR]` over the synopsis of Figure 2(b). After applying the Chain Rule and independence assumptions, the selectivity is expressed as $f(M/AR/IR[A/MR]) \approx f(AR/IR) \cdot f(IR[/A])$, where `AR/IR` and `IR/A` are the non-stable edges of the embedding. An XSKETCH summary would now employ Assumptions **A1** and **A2** to approximate the needed fraction terms; on the other hand, an fXSKETCH makes use of the corresponding fractional stabilities, arriving at the result $f(M/AR/IR[A/MR]) \approx \mathbf{B}_q(AR, IR) \cdot \mathbf{F}_q(IR, A)$. Note that fractional stabilities are at least as accurate as XSKETCH uniformity assumptions and, thus, assuming a fixed synopsis graph, the fXSKETCH estimate is guaranteed to be at least as accurate as the XSKETCH estimate (provided that independence is a valid assumption). Of course, for a fixed space budget, the fXSKETCH synopsis graph is typically smaller (see discussion above); nevertheless, our experimental results clearly show that, even in this case, our notion of fractional stabilities is a consistent winner.

Overall, the use of fractional stabilities simplifies the estimation framework by lifting the assumptions on backward- and forward-edge uniformity. In essence, the only assumptions needed by our selectivity-estimation algorithm are basic independence assumptions that de-correlate a selectivity fraction from other parts of the path embedding. The advantage of not applying uniformity assumptions is two-fold. First, estimates become more accurate since the required selectivity fractions are stored explicitly as fractional stabilities and need not be estimated (with potentially large errors). Second, the estimation process becomes faster, as applying uniformity typically entails scanning the parent (or, child) nodes of a synopsis node and computing sums of element counts.

The removal of the uniformity assumptions has a positive impact on the *synopsis-construction algorithm* as well. In essence, the BUILDXSKETCH algorithm is simplified since it only needs to consider `b-stabilize` and `f-stabilize` operations; the `b-split` refinement, which specifically targeted uniformity assumptions, becomes redundant and can be safely ignored. The end result is that the number of possible refinements per step is reduced, thus leading to faster synopsis-construction times.

5 Experimental Study

In this section, we present the results of an extensive experimental study on the performance of the new fXSketch model. The goal of this study is two-fold: (a) to evaluate the effectiveness of fXSketches as concise summaries for graph-structured XML data, and (b) to compare the accuracy of the new summarization model against the original XSketch framework. We have conducted experiments on real-life and synthetic XML data, using a variety of query workloads. Our key findings can be summarized as follows:

- The fXSketch synopses are effective summaries of graph-structured XML data, enabling accurate estimates for the selectivity of complex path expressions. The experiments show that, an XMark fXSketch synopsis achieves an average estimation error of 0.8%, with storage requirements that amount to 0.1% of the data size.
- fXSketches perform consistently better than XSketches in terms of estimation error. More specifically, higher accuracy is obtained for the same synopsis size, and a smaller size is achieved for the same accuracy. For instance, a 5KB fXSketch synopsis of the XMark data set has a 10 fold improvement in accuracy when compared to an XSketch synopsis of the same size. In addition, the fXSketch framework is more robust with respect to workloads that contain numerous branching predicates. For instance, the estimation error of a 5 KB XSketch synopsis for the IMDB data set can vary of up to 100%, between path expressions with and without branching predicates; on the other hand, an fXSketch of the same size has a variation of up to 15%.
- fXSketches compute relatively accurate estimates even for the coarsest synopsis. With XMark data set, the use of fractional stabilities in the coarsest summary have reduced the estimation error to 9% (compared to 27% for the coarsest XSketch synopsis).
- fXSketches have reduced requirements in terms of construction time. Given a specific space budget, an fXSketch is both more efficient to construct and more accurate compared to a XSketch. In addition, fXSketches provide accurate estimates even for low space budgets, thus leading to shorter construction times.

Overall, our experimental findings verify the effectiveness of the fXSketch framework and demonstrate its benefits over the original XSketch proposal.

5.1 Experimental Methodology

Implementation. We implemented a prototype of the proposed fXSketch framework over the existing XSketch code-base. Our implementation encodes fractional stabilities in the standard float representation, so each non **B**- of **F**-stable edge contributes an additional 4 bytes to the synopsis size. Note that XSketch and fXSketch synopses of the same size are different in the size of the synopes graphs - an fXSketch will always be the smaller graph (less nodes and edges).

The parameters of the build algorithm were set to V=10% and P=200 for both fXSketch and XSketch construction (details on the construction algorithm can be

found in the XSketch studies [5,6]). In the results that we report, the maximum size of the computed synopses was limited to 2% of the underlying XML data size.

Data Sets. We use two graph-structured data sets[1] in our experiments: IMDB, a real-life data set from the Internet Movie Database (`www.imdb.com`), and XMark, a synthetic data set that records the activities of an on-line auction site. Table 1 summarizes the main characteristics of the data sets in terms of the file size, and the sizes of the corresponding perfect and coarsest synopsis. The coarsest summary, termed the *label-split graph*, partitions elements to nodes based solely on their label. The perfect summary, termed the *B/F-Bisimilar graph*, partitions elements so that all the edges of the resulting synopsis are **B**- and **F**-stable (i.e., their fractional stabilities are equal to 1). This property guarantees that the selectivity estimate for any branching path expression has zero error. Overall, both data sets have large perfect summaries thus motivating the need for concise synopses. Note that the sizes reported do not include the space needed to store the actual text of the element labels; each label is hashed to a unique integer and the mapping is stored in a separate structure that is not part of the summary.

Table 1. Characteristics of the three data sets

	IMDB	**XMark**
File Size	3 MB	10 MB
Number of elements	102,755	206,131
Nodes in Label-Split Graph	123	84
Nodes in B/F-Bisimilar Graph	49,181	197,508
Size of Label-Split Graph	3.4 KB	2.1 KB
Size of B/F-Bisimilar Graph	1.1 MB	4.5 MB

Table 2. Average result sizes

	IMDB	XMark
Simple	484	1125
Light-Branching	1351	1773
Heavy-Branching	1331	2420

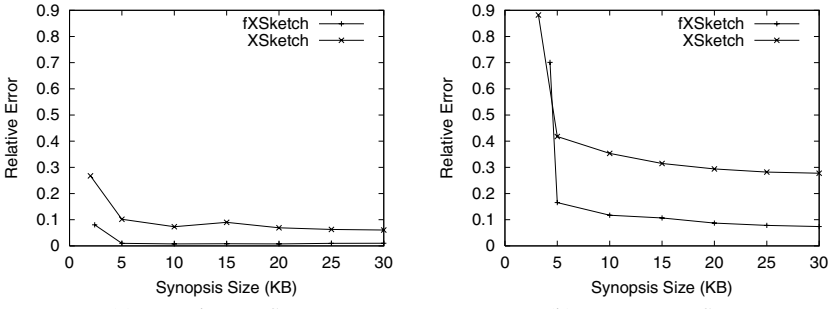
Query Workload. We evaluate the accuracy of the generated summaries against three different workloads, each one consisting of 1000 path expressions: (a) *Simple Paths*, which contains simple path expressions only, (b) *Heavy Branching Paths*, in which 90% of path expressions have branching predicates and (c) *Light Branching Paths*, in which 40% of path expressions have branching predicates. In each case, all path expressions are positive, i.e., they have non-zero selectivity, and are generated by sampling paths from the corresponding perfect synopsis. Except for branching predicates, which comprise of one or two steps, the length of the sampled paths is distributed between 2 and 5 and the sample is biased toward high counts in the perfect synopsis. As a result, the generated path expressions follow the distribution of the data, with high-count labels being referenced more frequently in the query set. Table 2 shows the average result size (in terms of the number of elements) for the path queries in each workload.

We have also experimented with *negative* workloads, i.e., path expressions that do not discover any elements in the data graph. Our results have shown that both XSketch

[1] We use the same data sets as the original XSketch study [5]

and fXSketch summaries consistently produce close to zero estimates with negligible error and therefore we omit this workload from our presentation.

Evaluation Metric. As in the original XSketch study [5], we quantify the accuracy of both XSketches and fXSketches based on the average absolute relative error of result estimates over path expressions in our workload. Given a path expression p with true result size c, the absolute relative error of the estimated count e is computed as $|e - c| / max(c, s)$. Parameter s represents a sanity bound that essentially equates all zero or low counts with a default count s and thus avoids inordinately high contributions from low-count path expressions. We set this bound to the 10-percentile of the true counts in the workload (i.e., 90% of the path expressions in the workload have a true result size $\geq s$).



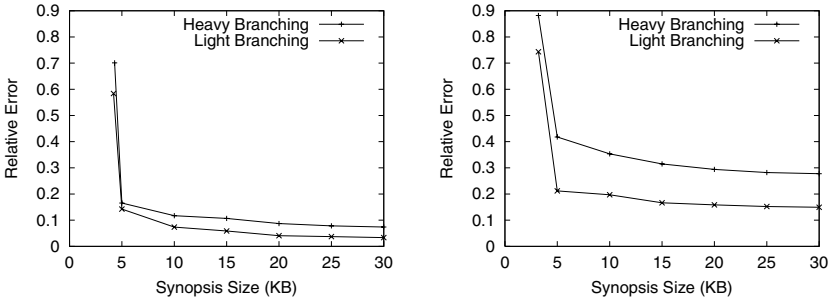
(a) XMark Data Set (b) IMDB Data Set

Fig. 4. XSketch and the fXSketch estimation error for Heavy Branching Paths

5.2 Experimental Results

fXSketch Performance for Branching Paths. In this experiment, we evaluate the performance of our fXSketch synopses for the heavy-branching paths workload. Figure 4 depicts the estimation error of fXSketches and XSketches as a function of the synopsis size, for the IMDB and XMark data sets. Note that, in all the graphs that we present, the estimation error at the smallest summary size corresponds to the label-split graph synopsis (i.e., the coarsest summary). Overall, the results indicate that fXSketches are effective summaries that enable accurate estimates for the selectivity of branching path expressions. In the case of XMark, for instance, the estimation error is reduced below 1% after a few refinements and it remains stable thereafter. For the more irregular IMDB data set, the estimation error stabilizes at 7% for a space budget of 25KB, which represents a very small fraction of the original data size. Compared to XSketches, it is evident that the new fXSketch synopses perform consistently better. For the IMDB data set, for instance, a 25KB fXSketch has an average error of 7%, compared to 28% for the XSketch of the same size - a 4 fold improvement. It is interesting to note that the fXSketch-computed estimates are significantly more accurate even for the case of the coarsest synopsis. For the XMark data set, for instance, the average error for the coarsest

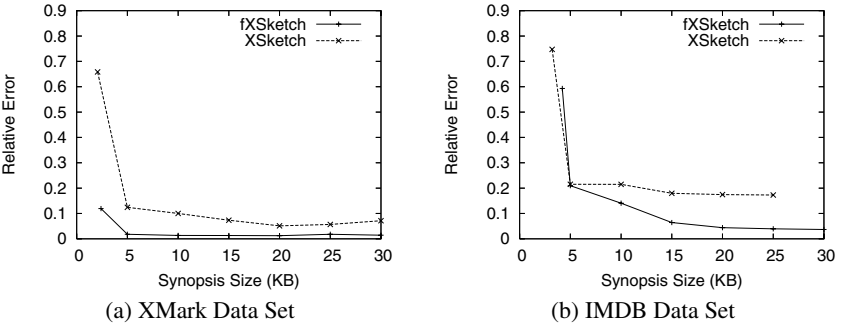
fXSketch is 8% (2.5KB of storage), while the error for the coarsest XSketch is more than 25% (2.1KB of storage).



(a) fXSketches (b) XSketches

Fig. 5. Estimation accuracy for different branching worklads: (a) fXSketches, (b) XSketches.

Workload Comparison. Figure 5 depicts the estimation error for XSketch and fXSketch synopses respectively, for the two different types of branching workloads (we note that the numbers for the Heavy Branching workload are identical to Figure 4). We report the results for the IMDB data set only, as its structure is more irregular than that of the XMARK data set. The plot indicates that fXSketches are more robust in terms of their performance than XSketches. As shown in Figure 5(b), the fXSketch error follows a similar pattern in both types of workload, with the error of Heavy Branching being slightly increased (as expected). XSketches, on the other hand, exhibit significantly different errors depending on the workload type (Figure 5(a)). The increased error for Heavy Branching suggests that the forward-edge uniformity assumption is not valid in the underlying data graph, thus leading to significant errors as the number of branches increases. fXSketch estimation, on the other hand, relies on fractional stabilities in order to capture accurately the distribution of document edges along each non-stable synopsis edge; as a result, an fXSketch summary provides better approximations for smaller budget sizes.



(a) XMark Data Set (b) IMDB Data Set

Fig. 6. XSketch and the fXSketch estimation error for Simple Paths

fXSketch Performance for Simple Paths. In this experiment, we evaluate the performance of fXSketches for simple path expressions. Figure 6 depicts the estimation error of fXSketches and XSketches as a function of the synopsis size, for the IMDB and XMark data sets. Similar to the previous experiment, fXSketches provide more accurate estimates compared to XSketches. For the XMark data set, the estimation error for fXSketches stabilizes at 1.7% (5KB of storage), while for XSketches it remains at a considerably higher 7% (20KB of storage). The results follow a similar trend for the IMDB data set, where the error for fXSketches is reduced to 4% for the 20KB synopsis, while the XSketch error stabilizes at 17% for the same space budget. In both data sets, and in accordance to our findings in previous experiments, we observe an improvement in accuracy for the coarsest summaries. The difference is more notable in the XMark data set - 12% vs. 66% - and comes at a very small increase in the storage requirements of the summaries: 2450 bytes for the fXSketch vs. 2100 bytes for the XSketch.

6 Conclusions and Future Work

Estimating the selectivity of complex path expressions is a key step in the optimization of declarative queries over XML data. In this paper, we have proposed the fXSketch model, a generalization of the original XSketch framework to the new concept of fractional stabilities. Intuitively, fractional stabilities capture the degree of stability of synopsis edges, and essentially free the estimation framework from the, potentially, erroneous, uniformity assumptions. The net result is a simplified estimation framework that can provide more accurate estimates with less computation. Results from an extensive experimental study have verified the effectiveness of the new model in providing low-error selectivity estimates for complex path expressions and have demonstrated its benefits over the original XSketch synopses.

In our future work, we plan to fine-tune certain aspects of the proposed framework. More specifically, the current storage overhead of fractional stabilities can reach up to 30-45% of the total synopsis size, depending on the data set. We intend to investigate techniques of reducing this overhead, by selectively choosing which fractional stabilities to materialize. In essence, this would allow a hybrid model where both fractional stabilities and the uniformity assumptions are used during estimation. A second direction that we intend to explore is the incremental maintenance of fXSketch synopses in the presence of data updates, or the refinement of summaries based on query feedback (self-tuning synopses). We believe that fractional stabilities are a suitable model for such techniques since they record distribution information at a finer level of detail and can thus track more reliably the statistical characteristics of the underlying data.

References

1. Clark, J., DeRose, S.: "XML Path Language (XPath), Version 1.0". W3C Recommendation (available from `http://www.w3.org/TR/xpath/`) (1999)
2. Aboulnaga, A., Alameldeen, A.R., Naughton, J.F.: "Estimating the Selectivity of XML Path Expressions for Internet Scale Applications". In: Proceedings of the 27th Intl. Conf. on Very Large Data Bases. (2001)

3. Freire, J., Haritsa, J.R., Ramanath, M., Roy, P., Siméon, J.: "StatiX: Making XML Count". In: Proceedings of the 2002 ACM SIGMOD Intl. Conf. on Management of Data. (2002)
4. Lim, L., Wang, M., Padmanabhan, S., Vitter, J., Parr, R.: XPathLearner: An On-Line Self-Tuning Markov Histogram for XML Path Selectivity Estimation. In: Proceedings of the 28th Intl. Conf. on Very Large Data Bases. (2002)
5. Polyzotis, N., Garofalakis, M.: "Statistical Synopses for Graph Structured XML Databases". In: Proceedings of the 2002 ACM SIGMOD Intl. Conf. on Management of Data. (2002)
6. Polyzotis, N., Garofalakis, M.: "Structure and Value Synopses for XML Data Graphs". In: Proceedings of the 28th Intl. Conf. on Very Large Data Bases. (2002)
7. Wang, W., Jiang, H., Lu, H., Yu, J.X.: Containment join size estimation: Models and methods. In: Proceedings of the 2003 ACM SIGMOD Intl. Conf. on Management of Data. (2003)
8. Wu, Y., Patel, J.M., Jagadish, H.: "Estimating Answer Sizes for XML Queries". In: Proceedings of the 8th Intl. Conf. on Extending Database Technology (EDBT'02). (2002)
9. Kaushik, R., Shenoy, P., Bohannon, P., Gudes, E.: "Exploiting Local Similarity for Efficient Indexing of Paths in Graph Structured Data". In: Proceedings of the Eighteenth Intl. Conf. on Data Engineering, San Jose, California (2002)
10. Milo, T., Suciu, D.: "Index structures for Path Expressions". In: Proceedings of the Seventh International Conference on Database Theory (ICDT'99), Jerusalem, Israel (1999)
11. Bray, T., Paoli, J., Sperberg-McQueen, C.M., Maler, E.: "Extensible Markup Language (XML) 1.0 (Second Edition)". W3C Recommendation (available from `http://www.w3.org/TR/REC-xml/`) (2000)
12. DeRose, S., Maler, E., Orchard, D.: "XML Linking Language (XLink), Version 1.0". W3C Recommendation (available from `http://www.w3.org/TR/xlink/`) (2001)
13. McHugh, J., Widom, J.: "Query Optimization for XML". In: Proceedings of the 25th Intl. Conf. on Very Large Data Bases. (1999)
14. Chamberlin, D., Clark, J., Florescu, D., Robie, J., Siméon, J., Stefanescu, M.: "XQuery 1.0: An XML Query Language". W3C Working Draft 07 (available from `http://www.w3.org/TR/xquery/`) (2001)
15. Paige, R., Tarjan, R.E.: "Three Partition Refinement Algorithms". SIAM Journal on Computing **16** (1987)