

Sketching Probabilistic Data Streams

Graham Cormode
AT&T Labs—Research
180 Park Avenue, Florham Park NJ
graham@research.att.com

Minos Garofalakis
Yahoo! Research and UC Berkeley
2821 Mission College Blvd, Santa Clara CA
minos@yahoo-inc.com

ABSTRACT

The management of uncertain, probabilistic data has recently emerged as a useful paradigm for dealing with the inherent unreliabilities of several real-world application domains, including data cleaning, information integration, and pervasive, multi-sensor computing. Unlike conventional data sets, a set of probabilistic tuples defines a probability distribution over an exponential number of *possible worlds* (i.e., “grounded”, deterministic databases). This “possible worlds” interpretation allows for clean query semantics but also raises hard computational problems for probabilistic database query processors. To further complicate matters, in many scenarios (e.g., large-scale process and environmental monitoring using multiple sensor modalities), probabilistic data tuples arrive and need to be processed in a streaming fashion; that is, using limited memory and CPU resources and without the benefit of multiple passes over a static probabilistic database. Such probabilistic data streams raise a host of new research challenges for stream-processing engines that, to date, remain largely unaddressed.

In this paper, we propose the first space- and time-efficient algorithms for approximating complex aggregate queries (including, the number of distinct values and join/self-join sizes) over probabilistic data streams. Following the possible-worlds semantics, such aggregates essentially define probability distributions over the space of possible aggregation results, and our goal is to characterize such distributions through efficient approximations of their key moments (such as expectation and variance). Our algorithms offer strong randomized estimation guarantees while using only sublinear space in the size of the stream(s), and rely on novel, concise streaming sketch synopses that extend conventional sketching ideas to the probabilistic streams setting. Our experimental results verify the effectiveness of our approach.

Categories and Subject Descriptors:

E.1 [Data]: Data Structures; F.2 [Theory]: Analysis of Algorithms

General Terms: Algorithms, Performance, Reliability

Keywords: Data Streams, Uncertain Data.

Work done while at Intel Research, Berkeley

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGMOD’07, June 12–14, 2007, Beijing, China.

Copyright 2007 ACM 978-1-59593-686-8/07/0006 ...\$5.00.

1. INTRODUCTION

Conventional database systems and query processing tools are designed around the idea of *static* collections of *exact* data tuples. Unfortunately, the data generated by a diverse set of real-world applications is often *uncertain and imprecise*. For instance, data integration and record linkage tools can generate distinct degrees of confidence for output data tuples (based on the quality of the match for the underlying entities) [8]; structured information extractors typically assign different confidences to rules for identifying meaningful patterns in their (unstructured) input [17]; and, pervasive multi-sensor computing applications need to routinely handle noisy sensor/RFID readings [19]. Motivated by these new application requirements, recent research efforts on *probabilistic data management* aim to incorporate uncertainty and probabilistic information as “first-class citizens” of the database system.

Among the different approaches for managing uncertainty inside the database, *tuple-level uncertainty models* (that, essentially, associate independent existence probabilities with individual tuples) have seen wide adoption both in research papers as well as system prototypes. This is due to both their simplicity of representation in current relational systems (i.e., just adding an “existence probability” column), as well as their simple and intuitive query semantics. In a nutshell, a probabilistic database is a concise representation for a probability distribution over an exponentially-large collection of *possible worlds*, each representing a possible “grounded” (deterministic) instance of the database (e.g., by flipping appropriately-biased independent coins to select each uncertain tuple). This “possible-worlds” semantics also implies clean semantics for queries over a probabilistic database — essentially, the result of a probabilistic query defines a probability distribution over the space of possible query results across all possible worlds [8].

Unfortunately, despite its simple, intuitive semantics, the paradigm shift towards tuple-level uncertainty also appears to imply a huge jump in the computational complexity of simple query processing operations: As demonstrated by Dalvi and Suciu [8], correctly evaluating the resulting tuple probabilities for duplicate eliminating projections over a simple three-way join can give rise to problems with $\#P$ -hard data complexity (i.e., exponential in the number of database tuples). Query-processing efficiency issues are, of course, further exacerbated by the *streaming nature* of several target applications for probabilistic data-management tools. For example, large-scale process and environmental monitoring tools rely on the continuous collection and processing of large amounts of noisy, uncertain readings from numerous sensor modalities, and recent work has already suggested attaching tuple-level probabilities as explicit indicators of data quality [19]. Due to the continuous, online nature of these applications and the large data volumes involved, these probabilistic data tuples arrive continuously and need to be pro-

