Approximate DataGuides

Roy Goldman, Jennifer Widom
Stanford University
{royg,widom}@cs.stanford.edu
www-db.stanford.edu

Abstract

DataGuides are concise and accurate summaries of semistructured databases, enabling schema exploration and improving query processing. Unfortunately, DataGuides can be very expensive to compute, especially for large, cyclic databases. For many DataGuide uses, an "approximate" summary of the database's structure can be beneficial yet much cheaper to compute. We summarize several uses of DataGuides and define *Approximate DataGuides (ADGs)*, which relax certain aspects of the DataGuide definition. An ADG allows some inaccuracy yet retains properties that make it useful in numerous situations. The core of the paper presents two general approaches for building ADGs, describing algorithms and experimental results.

1 Introduction

A *DataGuide* is a concise and accurate structural summary of a semistructured database. Originally proposed in [GW97], DataGuides have a variety of uses that enable query formulation and processing in a semistructured database management system [MAG+97]:

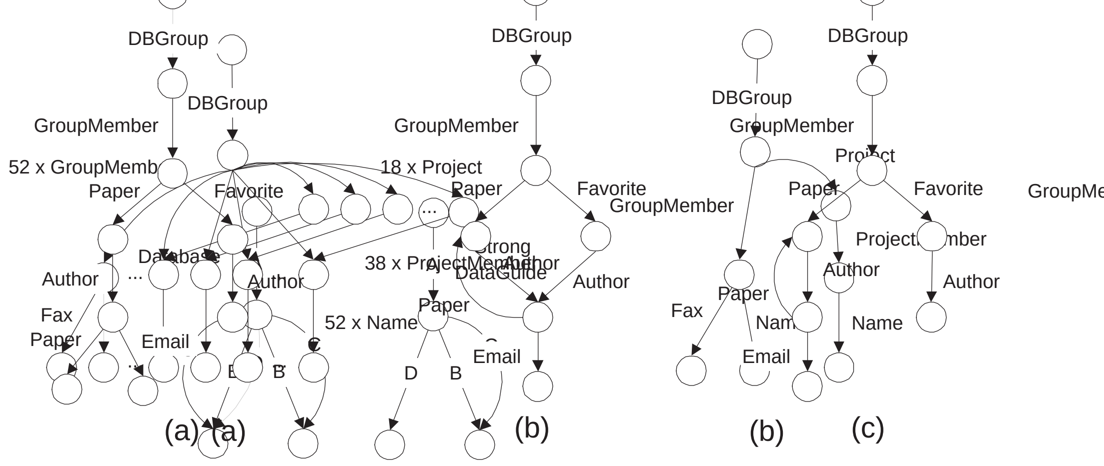
- *UI*: as a user interface to help users explore database structure and submit queries "by example"
- *Statistics*: to store statistics about the shape of the database, for use by a query optimizer [MW97]
- *Warnings*: as a tool to provide warnings about queries that refer to nonexistent paths
- *Path Expressions*: to enable compile-time expansion of regular path expressions [MW98]
- *Path Index*: as a path index to speed up query processing [GW97]

Semistructured databases usually are modeled as labeled, directed graphs [Abi97, Bun97]. For a tree-shaped database, DataGuide construction is linear in space and time with respect to the size of the database. For a general graph, however, the algorithm is exponential in the worst case. For many graph-structured databases, experiments show that DataGuides can be computed quickly and are much smaller than the original database [GW97]. Still, we have seen examples—particularly cyclic databases—where DataGuide construction is prohibitively expensive. In this paper we propose *Approximate DataGuides (ADGs)*. By relaxing the definition of a DataGuide, we can provide many of the benefits of a DataGuide yet avoid the associated performance traps.

We present our work in the context of *OEM* [PGMW95], a popular model for semistructured data where a database is a rooted, directed graph, with textual labels on edges and atomic values in leaves. A DataGuide is itself an OEM graph G that corresponds to an OEM database D, such that every distinct label path from the root of D appears exactly once as a path from the root of G, and every label path from the root of G has at least one matching label path in D.

Quite simply, an ADG drops the second requirement that all DataGuide paths must exist in the original database. Therefore an ADG may have "false positives" but never "false negatives" concerning the existence of database paths. Reconsider the five DataGuide uses above:

- *UI*: A user exploring an ADG may see paths that do not actually exist in the database.
- *Statistics*: We can still associate statistics with every ADG object, hence we can store statistics for every rooted path in the database. However, some statistics may be based on a superset of the actual objects reachable along that path.
- *Warnings*: The system may fail to warn the user that certain path expressions do not exist, but it will never incorrectly warn that a valid path does not exist.
- *Path Expressions*: Expanding to a superset of valid path expressions is not harmful, although it can degrade efficiency [MW98].
- *Path Index*: An index is expected to be exact, so an ADG is not usable.


DBGroup
GroupMember
52 x GroupMember
Paper
Favorite
18 x Project
Database
Author
Fax
Email
38 x ProjectMember
52 x Name
Strong DataGuide
D
B
C
(a)
(b)
(c)
Project
ProjectMember
Name