Considering User Intention in Differential Graph Queries

Elena Vasilyeva, SAP SE, Dresden, Germany
Maik Thiele, Database Technology Group, Technische Universität Dresden, Dresden, Germany
Christof Bornhövd, SAP Labs, LLC, Palo Alto, CA, USA
Wolfgang Lehner, Database Technology Group, Technische Universität Dresden, Dresden, Germany

ABSTRACT

Empty answers are a major problem by processing pattern matching queries in graph databases. Especially, there can be multiple reasons why a query failed. To support users in such situations, differential queries can be used that deliver missing parts of a graph query. Multiple heuristics are proposed for differential queries, which reduce the search space. Although they are successful in increasing the performance, they can discard query subgraphs relevant to a user. To address this issue, the authors extend the concept of differential queries and introduce top-k differential queries that calculate the ranking based on users’ preferences and significantly support the users’ understanding of query database management systems. A user assigns relevance weights to elements of a graph query that steer the search and are used for the ranking. In this paper the authors propose different strategies for selection of relevance weights and their propagation. As a result, the search is modelled along the most relevant paths. The authors evaluate their solution and both strategies on the DBpedia data graph.

Keywords: Differential Queries, Flooding, Graph Databases, Result Ranking, Routing, Top-K Queries

INTRODUCTION

Following the principle “data comes first, schema comes second”, graph databases allow to store data without having a predefined, rigid schema and enable a gradual evolution of data together with its schema. Unfortunately, schema flexibility impedes the formulation of queries. Due to the agile flavor of integration and interpretation processes, users very often do not possess deep knowledge of the data and its evolving schema. As a consequence, issued queries might return unexpected result sets, especially empty results. To support users to understand the reasons of an

DOI: 10.4018/JDM.2015070102
empty answer, we already proposed the notion of a differential query (Vasilyeva, Thiele, Bornhövd, & Lehner, 2014), a graph query (for example see Figure 1(a)) which result consists of two parts:

1. A discovered subgraph that is a part of a data graph isomorphic to a query subgraph like in Figure 1(b),
2. A difference graph reflecting the remaining part of a query like in Figure 1(c).

Although our approach (Vasilyeva, Thiele, Bornhövd, & Lehner, 2014) already supports users in the query answering process, it still has some limitations: the number of intermediate results can be very large, e.g., it can reach up to 150K subgraphs for a data graph consisting of 100K edges and a graph query with 10 edges. A user has two alternatives to deal with so many answers. The results can be ranked based on a scoring function, but the size of the result set stays the same; as a consequence, the response time can be very high. Alternatively, the results can be pruned by optimization strategies that reduce the number of traversals for a query based on cardinality and degree of a query’s vertices. This approach reduces significantly the response time. As a negative side effect, optimization strategies may remove important subgraphs, since these strategies do not consider the users’ intention.

CONTRIBUTIONS

To cope with this issue, we extend the concept of differential queries with a top-k semantic, resulting in so-called top-k differential queries (Vasilyeva, Thiele, Bornhövd, & Lehner, 2014). These queries allow the user to mark vertices, edges, or entire subgraphs of a graph query with relevance weights showing how important graph elements are in a query. To make the search of a top-k differential query with multiple relevance weights possible, we present two algorithms for the propagation of relevance weights: relevance flooding and broadcasting. Based on the propagated weights, the system decides automatically how to conduct the search in order to deliver only the most relevant subgraphs to a user as an alternative result set of the original query. The initial weights are used to rank the results. The concept of top-k differential queries allows us to reduce processing efforts and to rank individual answers according to the user’s interest.
Selecting a Database Partitioning Technique
[www.igi-global.com/article/selecting-database-partitioning-technique/51123?camid=4v1a](www.igi-global.com/article/selecting-database-partitioning-technique/51123?camid=4v1a)

The Schema Mapper: An Expert System that Determines the Least Cost Physical File Structure in a Database Management System
[www.igi-global.com/article/schema-mapper-expert-system-determines/51177?camid=4v1a](www.igi-global.com/article/schema-mapper-expert-system-determines/51177?camid=4v1a)

Roles of Resource and Data Contention on the Performance of Replicated Distributed Database Systems
[www.igi-global.com/article/roles-resource-data-contention-performance/51115?camid=4v1a](www.igi-global.com/article/roles-resource-data-contention-performance/51115?camid=4v1a)