**POLYTICS:** Provenance-Based Analytics of Data-Centric Applications

Pierre Bourhis  
CNRS, UMR 9189 - CRISTAL  
Daniel Deutch  
Tel Aviv University  
Yuval Moskovitch  
Tel Aviv University

**Abstract**—We consider in this demonstration the analysis of complex data-intensive applications. We focus on three classes of analytical questions that are important for application owners and users alike: Why was a result obtained? What would be the result if the application logic or database is modified in a particular way? How can one interact with the application to achieve a particular goal? Answering these questions efficiently is a fundamental step towards optimizing the application and its use. Noting that provenance was a key component in answering similar questions in the context of database queries, we have developed POLYTICS, a system that employs novel provenance-based solutions for these analytic questions for data-centric applications. We propose to demonstrate POLYTICS using an online bicycle shop application as an example, letting participants play the role of both analysts and users.

Video: https://youtu.be/mOBpUh7luO4

I. INTRODUCTION

Our proposed demonstration focuses on the analysis of complex applications that rely on, and dynamically update, an underlying database in the course of their execution. The complexity of such applications leads to many challenges, faced by both the application owners and users. The owner needs to analyze the application and logs of its executions, so that she can identify bugs and misuses and ultimately optimize the application; the user typically wishes to identify optimal uses of the application. We start by presenting the example used throughout our demonstration. It involves an online bicycle shop allowing users to view bicycles, parts and accessories, and choose products to add to the shopping cart. Upon item selection, the system updates its price according to the availability of discount deals. Before payment, the user can remove products from the shopping cart, and if the order is not empty she can pay and exit. We now present the main scenario for our analysis.

**Example 1.1:** Consider a user who first adds a mountain bike to the shopping cart, and then a cycling helmet. Assuming a discount deal for combined bike and helmet purchase, the helmet price is updated to a discount price. Then, before payment, the user removes the bike from the shopping cart. Due to a bug in the application logic, the helmet price may remain as if the discount applies, although eventually the user has not purchased the mountain bike. Upon viewing the wrong price, multiple questions arise: why was it obtained as such? what would be the price if the owner changed the database / application logic in a particular way? how to interact with the system to obtain a correct price?

We model the online shop as a data-centric process whose partial state machine and underlying database are shown in Figures 2 and 1 respectively. Transitions are associated with insertion/deletion/modification queries, in turn captured by (union of) Conjunctive Queries augmented with a +/-M sign for insertion/deletion/modification. The database includes a relation for each item type, a Deals relation for special offers, a Cart relation storing the products selected by the user, and relations standing for user input ($R_u$, $R_b$, $R_a$ and $R_c$) in different transitions. The formal model appears in [1].

In the context of database queries, a prominent approach (see e.g. [2], [3], [4]) for analyzing answers is based on the tracking of provenance, i.e. a record of the transformations that data undergoes. The idea is to efficiently track the “core” aspects of the transformations that have taken place, and then use it for answering questions such as the above. Such solution is absent in the context of data-centric applications. To address this need, we have implemented POLYTICS, PrOvenance-based ANALYTICS for data-centric applications. POLYTICS leverages the model and algorithms for provenance generation and usage from [1]; we next briefly highlight the role of provenance in answering analytical questions.

**Example 1.2:** Figure 1 depicts a database fragment with provenance annotations next to tuples. Intuitively, the provenance of an output tuple (in this example, tuples of the Cart table) describes the relevant actions leading to it; in the case of (Helmet, $\$25$) these are $p_1$ and $p_3$, which are associated in the state machine with insertion of bicycles and accessories (resp.) to the shopping cart. It also includes relevant database tuples, such as the mountain bike ($d_1$) and the helmet ($d_3$), as well as the existence of a deal ($d_4$) and the user choices ($u_1, u_2$). Importantly, it also shows the way in which they are combined to form the output (in this case via conjunction). This is highly useful for explaining a result tuple (i.e. answering a “why” question): such explanation would contain only events that have affected the result, along with the relevant data items. In the above example, an explanation to the helmet price would be the insertion of bike and helmet to the shopping cart, and the deal on the helmet; the provenance expression “translates” into such intuitive explanation. Furthermore, provenance is useful for “what-if” analysis, where every hypothetical scenario corresponds to a boolean assignment to the provenance annotations. For instance, assigning false to $d_1$ and true to the other variables corresponds to the scenario where the mountain bike are not available. The shopping cart in this case would contain only the helmet, showing a price of $50 (instead of $25). Last, for “how-to?” queries, a more complex provenance expression is generated, capturing the set of all possible executions rather than a particular one; a SAT solver is then used to find an execution yielding a tuple or sub-instance
of interest, such as a desired helmet price.

**Related Work:** The use of data provenance for “why” (e.g. [2]), “what-if” (e.g. [3]) and “how-to” (e.g. [4]) has been extensively studied, focusing on database queries rather than on data-centric processes. In [5] we have proposed a “what-if” analysis of data-centric processes; POLYTICS supports a significantly larger class of applications (specifically, ones that can update the underlying database) and of analysis questions (including “why” and “how-to”).

**II. SYSTEM OVERVIEW**

POLYTICS’s server side is implemented in C#, and client side in Angular JS using Bootstrap framework. The client web application is deployed on Node.js JavaScript runtime environment and runs on Windows 10. The system architecture is depicted in Figure 3. The system requires the application’s owner to provide a description of the application, including an FSM describing its flow and its database. Each action in the FSM and each DB relation should further be associated with a textual description (for presentation purposes). POLYTICS may be used both by system analysts and users. For “what-if” and “why” analysis, users/analysts interact with a dedicated interface; for “how-to” analysis they interact with a wrapper of the original application allowing them to view recommendations for navigation. We next explain the main system components.

**Provenance engine:** The provenance engine consists of two generators: (1) real-time generator, that tracks the provenance of executions, and is used for “why” and “what-if” analysis; (2) static generator, that computes, based on the application structure, a provenance expression capturing the set of all possible executions, used for “how-to” analysis.

**Analysis Interface:** Provenance is fed to the analysis engine whose output is demonstrated in Fig. 4 (the “how-to” screen is omitted for lack of space). The “why” interface allows to choose an output tuple and view a textual representation of its reason, based on the provenance and on the textual description provided for each action and DB relation. The “What-if” screen allows to apply hypothetical modifications and observe, in an interactive speed, their effect on the output.

**III. DEMONSTRATION SCENARIO**

We will demonstrate the usefulness of POLYTICS in the context of a simple online bicycle shop as described above.

![Fig. 1: Database with provenance](image1)

![Fig. 2: Partial Process Logic](image2)

![Fig. 3: System architecture](image3)

![Fig. 4: Analysis output](image4)