DANCE: Data Cleaning with Constraints and Experts

Ahmad Assadi
Tel Aviv University

Tova Milo
Tel Aviv University

Slava Novgorodov
Tel Aviv University

***Abstract*—We demonstrate DANCE, a system that assists domain experts in the efficient resolution of integrity constraints violation. DANCE is demonstrated on the UEFA Champions League database, employing the ICDE'17 audience for effective data cleaning.**

Video: https://youtu.be/IssLjM2WQQQ

I. Introduction

Data cleaning is a long standing problem that has attracted much research interest in the past years in the databases community. Popular techniques for data cleaning use data-integrity and consistency rules to identify errors in the data and to automatically resolve them, e.g. by fining a *minimal repair* that will resolve the constraints violation [1], or by using predefined *priorities* among possible resolutions [2]. Such automatic solutions however cannot ensure precision of the repairs since they do not have enough evidence about the actual errors and thus may in fact lead to wrong results with respect to the ground truth. In order to overcome the limitations of such automatic techniques it has been suggested to use *domain experts* that have extensive knowledge about the ground truth, to examine the potential updates and choose which should be applied to the database [2], [3], [4], [5]. However, the sheer volume of the databases and the large number of possible updates that may resolve a given constraint violation, may make such a manual examination prohibitory expensive. The goal of the DANCE prototype system presented here is to help optimizing the experts work and reduce as much as possible the number of questions (updates verification) they need to address. As we will describe, our algorithms effectively prune the search space and minimize the amount of interaction with the experts while, at the same time, maximize the potential "cleaning benefit" derived from the experts' answers. DANCEcan be used to optimize the initial cleaning of a databases as well as assist in its ongoing maintenance - whenever an constraint violation is reported, DANCEcan take over to efficiently clean the underlying database by interacting with the experts.

We will demonstrate DANCE over a UEFA Champions League database, employing the ICDE'17 audience for the effective resolution of real-life integrity constraints violation. We will show how DANCE's underlying optimization mechanism accelerate the data cleaning process and minimizes the number of questions that need to be posed to the crowd.

II. Technical background, in brief

Given a constraint violation, our algorithm first identifies the tuples in the database whose update may contribute (directly or indirectly) to the constraint resolution. We call those *suspicious tuples*. Database constraints may be inter-related and thus when analyzing a constraint violation these relationships must be taken into consideration. To determine which tuples should be considered first, we examine for each tuple t (1) the potential effects of updates to t, namely what tuples may potentially become unsuspicious if t is found to be incorrect and correspondingly updated/removed, (2) the number of potential updates (attribute errors) to t that may lead to such an effect, and (3) the probability, if known, for errors in the database relation to which t belongs. Using this information we build a graph whose nodes are the suspicious tuples and whose weighted edges capture the likelihood of an error in one tuple to occur and effect the other. Page-rank style algorithm is then used to identify the most beneficial tuples to ask about first.

Example: To illustrate let us consider the following simple example. The database in Figure 1 shows a portions of UEFA Champions League 2016/17 statistics database. The dark gray rows represent wrong tuples and lightgray rows represent missing tuples. The *Games* relation describes the results of a match between two teams, it stores the teams name, goals score and the stage. The *Teams* relation describes a football team, it stores the team name and country. The *Countries* relation describes the name of the country and number of teams that advanced to the group stage. We consider in our work integrity constraints described by standard tuple-generating and equality-generating dependencies [6]. The following two integrity constraints are relevant to this database: (i) two teams from the same country cannot play against each other on a group stage (ii) if country has at least one representative, its team must appear in the teams table. These are captured by the following rules.

- $Games(x_1, x_2, x_3, x_4, x_5) \wedge x_5 = "GroupStage" \wedge Teams(x_1, y_1) \wedge Teams(x_2, y_2) \rightarrow y_1 \neq y_2$
- $Countries(x_1, x_2) \wedge x_2 > 0 \rightarrow Teams(y_1, x_1)$

We assume that all the given rules are correct and reflect the ground truth. In our running example, as well as in the demo, the rules are derived from UEFA official regulation. Since the database is aggregated from multiple sources it contains mistakes and violates some of the constraints. One can notice for instance that the database mistakenly associates both the Celtic and the Manchester City football clubs to United Kingdom. However, despite the fact that Celtic and Manchester City are actually located in the United Kingdom they belong to distinct federations (that represent Scotland and

Games

team1	team2	t1_goals	t2_goals	stage
Celtic	Manchester City	3	3	Group Stage
Celtic	Hapoel Beer Sheva	5	2	Qualification

Teams

name	country
Celtic	UK
Manchester City	UK
Hapoel Beer Sheva	Israel
CSKA Moscow	Russia

Countries

name	num_of_teams
Israel	0
UK	5
England	4
Scotland	1

Fig. 1: Sample of UEFA Champions League DB

England separately), hence can play against each other.

When applying the integrity constraints to the database, we discover several inconsistencies. Each such inconsistency involves several tuples that when assigned together to the atoms in the body of the rule yielded a constraint violation. For example, a violation to the first rule involves a set of three tuples: *Games(Celtic, Manchester City, 3, 3, Group Stage)*, *Teams(Manchester City, UK)*, *Teams(Celtic, UK)*, whose existence in the database of lead to the violation. Intuitively, each of the tuples is *suspicious* and at least one is wrong and needs to be updated/deleted (otherwise the rule is incorrect which we assume is not the case). Also note that since the two constraints are inter-related, when a given tuple is suspicious other tuples become suspicious as well. Consider for example the second constraint, that requires that for each country in the *Countries* relation with a positive number of teams, there must be at least one team in *Teams* relation from this country. Relation *Countries* contains the tuple *(UK, 5)*, which enforces the existence of teams from United Kingdom. Since the *Teams(Celtic, UK)* and *Teams(Manchester City, UK)* tuples are *suspicious* (and may generally both be wrong), we may suspect also the tuple *Countries(UK, 5)*.

Which of these four suspicious tuples is more beneficial to consult about first with the expert? To determine this we build a directed graph whose nodes are the suspicious tuples and whose (weighted) edges capture the dependency between the suspicious tuples. Let β be the probability of an error in the relation R to which a tuple t belongs to. Intuitively, there is an edge from tuple s to t with a weight $n \times (1-\beta)$ if there are n attributes in t that one can change in order to eliminate at least one violating assignment that involves s. For example, data from official UEFA website will get a β close to 1, while user generated content in the other relations should get much less. We use 0.5 as default value. The graph for the four tuples that we obtain is depicted in Figure 2 (ignore for now the number on the nodes).

To decide which tuple to verify first, we process the graph using a PageRank-style [7] algorithm, to rank the nodes, and ask the experts about the nodes with highest rank. When answers are gathered, the database is updated accordingly, and incremental computation is applied to update the graph and identify the next candidates. The resulting ranks for our running examples are depicted on the nodes, and so we will ask about C (which is indeed incorrect and will be removed, instead *(England, 4)* and *(Scotland, 1)* will be inserted by the expert), $T1$ (incorrect, updated to *(Celtic, Scotland)*) and $T2$ (incorrect, updated to *(Manchester City, England)*). G is then no longer suspicious and no constraint is violated.

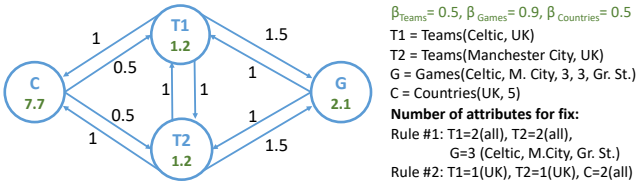

Fig. 2: Suspicious tuples graph

III. Demonstration Scenario

We will demonstrate the functionalities of DANCE on the UEFA Champions League database. Despite the fact that most data comes from a trusted sources (UEFA official web-site and related Wikipedia pages, official list of cities and countries), some cames from user generated content (sport news websites) as well as semi-automatic statistical tools, and so the integrated databases contains errors originating from the data generation and integration process. We will use UEFA official regulations as integrity constraints and employ the ICDE'17 attendees as experts for violations resolution.

During the demonstration we will explain the details and nuances of our system and discuss the decisions taken by the algorithm. In particular, we discuss why certain questions were chosen by the system, explain the effect that experts' answers have on the system's state, and show how the system infers when sufficient information has been collected to determine all the required corrective updates.

To stress the novelty of our solution and the effectiveness compared to other systems, we run in parallel competitor algorithms that do not use experts or alternatively employ them but do not optimize their usage, and show how the former may make mistakes whereas the latter ask many more questions to the experts.

Related Work As mentioned before, many data cleaning techniques has been proposed in the past. Experts have been used for data cleaning in multiple contexts. [5] introduced the idea of cleaning only a sample of data to obtain unbiased query results with confidence intervals. [2], [4] employ experts to resolve constraints violation which cannot be automatically resolved using their predefined priority rules, but do not optimize the experts exploration of the updates space. Closest to our work is [3] that optimizes the use of experts for pruning errors in query results, but ignores database constraints.

Acknowledgements Partially funded by the European Research Council under the FP7, ERC grant MoDaS #291071

References

[1] M. Yakout, L. Berti-Equille, and A. K. Elmagarmid, "Don't be scared: use scalable automatic repairing with maximal likelihood and bounded changes," in *SIGMOD*, 2013, pp. 553–564.

[2] F. Geerts, G. Mecca, P. Papotti, and D. Santoro, "The LLUNATIC data-cleaning framework," *PVLDB*, vol. 6, no. 9, pp. 625–636, 2013.

[3] M. Bergman, T. Milo, S. Novgorodov, and W. C. Tan, "Query-oriented data cleaning with oracles," in *SIGMOD*, 2015, pp. 1199–1214.

[4] M. Yakout, A. K. Elmagarmid, J. Neville, M. Ouzzani, and I. F. Ilyas, "Guided data repair," *PVLDB*, vol. 4, no. 5, pp. 279–289, 2011.

[5] J. Wang, S. Krishnan, M. J. Franklin, K. Goldberg, T. Kraska, and T. Milo, "A sample-and-clean framework for fast and accurate query processing on dirty data," in *SIGMOD*, 2014, pp. 469–480.

[6] R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa, "Data exchange: semantics and query answering," *TCS*, vol. 336, no. 1, 2005.

[7] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer Networks*, vol. 30, no. 1-7, pp. 107–117, 1998.