A Cost-Based Model and Effective Heuristic for Repairing Constraints by Value Modification

Paper presents 2 new data repairing algorithms which are called Greedy Repair and Greedy Repair FD First. The main difference between these 2 algorithms and other data repair algorithms is using Functional Dependencies (FD) and Inclusion Dependencies (IND) together. Authors believe that using FD and IND would increase the data repairing performance. They also introduce 2 performance enchantment methods for these algorithms. They are called Nearby Tuples and Queue and their main purpose is to reduce the redundant computation in Greedy algorithms. Also, authors try to find the minimal-cost repairs while using only modification and insertion techniques for repairs.

The introduction section is pretty good for readers that don't know the topic. There is only one example in the paper but it is sufficient for all the subtopics. The beginning part of the data repairing is similar to entity matching from the last week. The difference is, this paper only uses data modification or insertion methods due to deletion causes information lose. I'm not sure tuple weights introduced in this paper or not, but using weights for deciding which tuple to repair is a good trick for data repairing algorithm. Since they used both FD and IND, using Greedy Repair FD First make sense. Because, as they mentioned in the paper, while repairing the IND dependencies, algorithm can disrupt the FD dependencies. Also their distance comparison function for matching aware of both long and short strings and calculates its result according to this. So that, they can decide which tuple to change with higher accuracy. In the experiment they tested both algorithms on both synthetic data (TPC) and real data (DVD) and this method better for algorithm performance evaluation. The last thing is Nearby and Q techniques for performance increase. Both increase the algorithm's performance but authors can also combine these 2 methods to even further performance enhancement.

Paper doesn't mention how the weight for tuples values computed. Using them is very good but how they are getting weight values? Also, they mentioned that they only use data modification and insertion methods for repairing. However, section 4.1 shows that merging and modifying costs are equal (1) in the example. They mentioned inserting new tuples would create null values and they create SQL comparison problems. Paper doesn't emphasize when or how to use insertion method, it mostly focuses on modification part.

Some of the steps in algorithm could be better explained. For example, they could improve Example 2 in section 3.1 *Constraint Repairs*. It was hard to understand the concept for me. In experiment evaluation part, they only compare Greedy algorithms. I think using other data repairing methods for performance/accuracy comparison is necessary for all new algorithms research papers. Because, if we don't know the performance of the algorithm against others, why would anybody choose it? Also, some of the line graphs are hard to understand. Most of the lines intertwines and reader cannot understand which is which. They could use non-zero y-axis, so that graph can focus on the middle part. Figure 1 example is easy to understand but while reading the paper I have to scroll up a lot of times to see the tables. They could use small tuples when they mentioned the table tuples like t1. Lastly, I would like to see a system like Magellan for repairing tasks.

2019-02-03 Baran Kaya, kayab@mcmaster.ca, 400284996