PATSQL: Efficient Synthesis of SQL Queries from Example Tables with Quick Inference of Projected Columns

MohammadAmin Raeisi

Summary

This paper proposes a novel sketch-based technique for efficiently synthesizing SQL queries, including aggregations, nested queries, and window functions from I/O tables that the user provides as an example. Previous methods have some performance issues depending on table size, which caused the computational cost to grow exponentially with the increase in table columns. The critical insight to achieve efficiency is that the projection operator (i.e., Select keyword in SQL) in a program sketch can be lifted above other operators by applying transformation rules in relational algebra without changing the semantics of the program. This paper's algorithm is implemented in the PATSQL tool.

PATSQL takes input and output tables and hints for the constants used in predicates in the query as input, then synthesizes a program in its DSL that satisfies the I/O tables and uses the hints provided. Finally, PATSQL converts the synthesized program in DSL to SQL, which is beyond the scope of this paper.

Before converting I/O tables to the DSL program, it defined the DSL grammar and the sketches' grammar in DSL. To synthesize the DSL program, first, initialize a set of sketches S with a singleton having the sketch $Table(\square)$ or $Order(Table(\square), \square)$, depending on whether the output table is sorted or not. In each iteration, we retrieve a sketch s with the minimum size of the sketches in S. For each sketch s, it completes the all $'\square'$s in the sketch to find a program p. Then checks whether the output table is equal to the evaluation result of p or not. If the check succeeds, we return the program p as a result. Otherwise, we generate additional sketches from the sketch s. The algorithm repeats iterations until it finds a program p that satisfies the output table or the time limit exceeds.

Strengths

- They have implemented a simple UI that can quickly run PATSQL through live programming.
- The source code of the tool has been placed on GitHub.
- It has explained well the weakness of previous works like SCYTHE and the reasons for the need for the new tools.
- The text is evident, and the content is well conveyed.
- It has made a significant improvement over the previous state-of-the-art tool SCYTHE.
- It has explained the problem and the general solution well with the help of examples and figures.
- The details of the proposed algorithms are entirely and accurately explained.
- It has compared this method well with the previous approaches by presenting various benchmarks and charts.

Weaknesses

- It has repeated some concepts in the abstract and introduction a lot, such as improving the cost of construction, which could be due to the lack of enough words for these parts.
- It would have been better to move Figure 2 to the right side of the page since it is not mentioned on the left side to make it less distracting.
- It mentioned this point in the overview: "In general, the schemas of example tables given to a PBE tool should be the same as the schemas of the original tables stored in a database." it has provided an example to explain it, which I think was irrelevant or needed more explanation.
- At the end of section 3, it mentioned that its DSL supports 17 out of 20 SQL keywords. It would have been better if it had mentioned the three keywords that are not supported.

- In part 3, it would have been better if it defined the size of the sketch better and more precisely and explained why the size of the window differs from the others and why the size of others is the same.
- The conclusion section gave many explanations about the future direction. Extracting these explanations as a future works section would have been better.
- Including the pseudo-code of the *AssignTables*, *CompleteSketch*, and *ExpandSketch* algorithms would have been better.
- In section 4.2, it would have been better to clarify all four modes of the φ function and their differences with more examples and explanations.