[keywords: SQL, databases, program synthesis, potentially reinforcement learning]

[an incomplete set of technical skills: familiar with SQL, Java, experience with databases, like and good at using existing software tools]

In this summer project, we plan to automatically generate efficient SQL queries from simple input-output examples. This is quite similar to the following paper:

<https://scythe.cs.washington.edu/>

<https://scythe.cs.washington.edu/media/scythe-pldi.pdf>

Or this paper:

<https://squares-sql.github.io/>

<http://www.vldb.org/pvldb/vol13/p2853-orvalho.pdf>

These papers also aim to automatically synthesize SQL queries, however, their techniques don’t really care about the performance of the synthesized programs. This is not ideal, because users typically want to run these queries on large databases and performance matters a lot in those cases. Therefore, in our summer project, we want to develop techniques that can generate SQL programs that both satisfy the input-output examples as well as are performant.

So in this “warm-up” exercise, try your best to learn about these papers and tools. The general format of this exercise is to finish as many tasks as you can and write a short report to demonstrate your understanding. You can choose to work on the tasks in any order you want. In case your time is limited, you should prioritize quality over quantity. That is, feel free to finish a subset of these tasks and write a report summarizing your findings with high quality.

Task 1. Read papers.

Feel free to choose one of the following papers and try to understand it in depth.

Option 1. <https://scythe.cs.washington.edu/>

Option 2. <https://squares-sql.github.io/>

You are not expected to understand everything in the papers, however, you do want to focus your attention and try to answer a few important questions, such as:

* Try to understand the query language they are using. The language may not necessarily be the query language you learned in your course and may be more “domain-specific”.
* What are the key ideas of their technique?
* How do their algorithms work? Try to run their algorithms “conceptually” on concrete examples.

You may need to learn certain concepts along the way, such as SMT solving. You don’t need to dig too deep there, but be sure to at least get a high-level understanding of those concepts so they don’t block you from understanding the papers.

Task 2. Try their tools.

Depending on which paper you chose, try to use their tool as well. Both tools are available on Github.

Option 1. <https://github.com/Mestway/Scythe>

The author recently started to implement it using Python: <https://github.com/Mestway/Scythe2>

The implementation may not be complete though at this point.

Option 2: <https://github.com/squares-sql/SQUARES>

Consider running the tool on their benchmarks. For instance,

<https://github.com/Mestway/Scythe/tree/master/data> contains the benchmarks that are used to evaluate Scythe. <https://github.com/squares-sql/SQUARES/tree/master/tests-examples> contains benechmarks used to evaluate SQUARES.

Try to understand some of their benchmarks, and understand the results that are produced by the tool.

Task 3. Play with the PostgreSQL cost model.

PostgreSQL is a relational database management system. It has a built-in cost model:

<http://shiroyasha.io/the-postgresql-query-cost-model.html>

Try to learn about how to use this cost model. There is plenty of documentation online about this.

Once you know how to use the cost model, use it to estimate the cost of the queries generated by the synthesis tools. That is, given a benchmark (assuming the benchmark contains a “ground-truth” program Q\_gt), run the synthesizer on this benchmark, produce a synthesized query Q, get the costs for Q and Q\_gt, and compare their costs.

Try to identify some benchmarks where cost(Q) is significantly bigger than cost(Q\_gt).

You may need to “translate” the queries in their benchmarks into PostgreSQL, as well as the synthesized queries, since they may not be in the same language.

Task 4 (optional). Think about how to automatically generate SQL queries that have low cost with respect to the cost model.

Feel free to think about this, but this will be the main research question in this summer project.

Task 5. Write a report.

Put only important things in the report. Be concise. Avoid copy-pasting or rephrasing things from their papers to your report.

You should consider putting what follows in the report, but feel free to add whatever you think is important.

* What are the benchmarks that you see a significant gap between cost(Q) and cost(Q\_gt)? Why is there a big gap for those benchmarks?