Instructions to Replicate Experiments

This document instructs how to recreate the experiments discussed in the paper "Answering (Unions of) Conjunctive Queries using Random Access and Random-Order Enumeration" by Nofar Carmeli, Shai Zeevi, Christoph Berkholz, Benny Kimelfeld, and Nicole Schweikardt.

Libraries Required:

To recreate the experiments presented Boost and cmake are required. As well as all other standard tools needed to compile and run c++14 code (g++, make, and an IDE or text editor of your choice).

Preparing the database:

For the specific queries used in the paper, you may only use a database generated via the TPC-H *dbgen* tool (see here). After generation, place all your **.tbl* files in the subdirectory "*db*". Afterwards you may need to update the following files:

- *db/numlines.txt* – This file maps each table name to the number of lines in the table. Each entry is composed of a relational symbol (for instance, "_C_" or "_O_") on the left and the number of rows on the right. When updating the file, only change the number that corresponds to the different relations (not the symbols).
  This accelerates the process of building the relations in memory by pre-allocating the required number of lines. It is important to note that this is not mandatory and does change the result of the experiments since we omit this time from preprocessing (as mentioned in the paper). This process is simply meant to make the experiments run a little faster. To forgo this step, fill out zeros (0) in the file's right sides.
  Important: the file must still be present with an entry per relation.
- *CQs/filenames.*txt, *UCQs/filenames.txt, and mcUCQs/filenames.txt* – These three files locate each relation's *tbl* file. They need to be updated if you decide to change the name or location of a *tbl* file.
  Important: the paths are relative to where the code is run from. E.g., *CQs/Q0 or UCQs/UCQ2 etc.*
- *UCQs/UCQ1/samplesizes.txt* – This file is used when measuring the total time of $Q_7^S \cup Q_7^C$ (*UCQ1*) across different percentages. The 1st entry should indicate how much is 1% of your answers, the 2nd should equate to 5% etc. The reason is that the UCQ algorithm is incapable of counting the size of UCQ (nor is it required from an enumeration algorithm), hence we specify when to halt enumeration.
- *UCQs/UCQ1/thresholds.txt* – This file is used when measuring the time spent of rejection .vs. time spent of answers in *UCQ1*. Each entry is a threshold of how many answers to enumerate before counting a "lap", outputting the results of rejection time and answer time, and resetting the counters. In the file given each entry is 10% of answers.

With these files (possibly) updated you are ready to run each experiment.

Experiment 1: CQ Running Time

This section discusses how to recreate the experiment depicted in Section 7.3.1 (*CQ running time*) and Figure 1.

The experiment requires running $REnum(CQ)$ and $Sample(EW)$ on 6 different CQs to enumerate increasingly larger requests of answers.

In order to obtain the output of $REnum(CQ)$ on $\{Q_0, Q_2, Q_3, Q_7, Q_9, Q_{10}\}$ you may compile each file individually or use our script. To run a query individually, e.g., Q_0 do the following:

1. Go to CQs/Q_0
2. *Run "make enum_test"*
3. Change the file *sampleratios.txt* to your desired percentages
4. Run the resulting executable 3 times
5. Go to the directory *TestsOutput* to view the resulting report file (of all 3 runs)
6. Average out the results to obtain the corresponding column

Our script *runEnumTests* performs steps 1-4 for all the CQs given as parameters. To run it use the commad *./runEnumTests <Qs>*, e.g, *./runEnumTests Q0 Q2 Q3 Q7 Q9 Q10*.

In order to obtain the results of $Sample(EW)$ on the CQs, run it as the instructions mentioned in the public repository of Zhao et al. detail (see here). Each CQ has its own main file (e.g., *query0_main.cpp)* that needs to be compiled and run.

Important: be sure to use our version of the $Sample(EW)$ code (under "*SampleJoin")* as it contains our change of rejection sampling, i.e., rejecting previously seen answers.

Experiment 2: CQ Delay Analysis

This section details how to run the experiment mentioned in Section 7.3.2 (*CQ delay analysis)*. The instructions are exactly as those of Experiment 1 with some changes:

1. $REnum(CQ)$ code must be compiled with make command "*make timing_test"* (as opposed to "make" or "make *enum_test"*)
2. The script to use is *runTimingTests* (instead of *runEnumTests*)
3. To obtain the results of $Sample(EW)$, compile it with the preprocessor symbol TIMING.

Important: change no.3 will only work on the $Sample(EW)$ code we provided as we added a function that works like the regular enumeration function but measures the delay of each answer.

The result is a csv file in which each delay has been logged, in order to create the boxplots simply use python's matplotlib or any other graphing tool.

Experiment 3: UCQ Analysis

This section details how to perform the three UCQ experiments described in Section 7.3.3, and depicted in figures 4a, 4b, and 5. In the files $Q_7^S \cup Q_7^C$ is denoted as *UCQ1,* $Q_2^P \cup Q_2^S \cup Q_2^N$ is denoted as UCQ2, and $Q_A \cup Q_E$ is denoted as *UCQ3*.

Figure 4a – Total time

- To run $CQRA - permute$ on each CQ, refer to previous sections of this document and follow their instructions.
- To run $CQRA - repeat$ on each UCQ go to its folder under *UCQs*, run "*make ucq_test"*, run the executable three times, and average the results.
- To run $CQRA - default$ navigate to the UCQ's folder under *ProbabillisticCounting*, then run "make UCQ1(/2/3)_Main". Run the resulting executable three times and average the results.

- To run $CQRA - double$ navigate to the UCQ's folder under *DualSampling*, then run "make UCQ1(/2/3)_Main". Run the resulting executable three times and average the results.
- To run $MCRA - permute$ on each query: Go to its folder under *mcUCQs,* run "*make mcucq1(/2/3*)".
  Important: No need to run three times and calculate the average here, as it is done internally.

Figure 4b – Varying percentage

Important: Remember that we're only building and running UCQ1 ($Q_7^S \cup Q_7^C$) in this experiment.

- To run $CQRA - permute$ on each CQ, refer to the previous subsection of this document.
- To run $CQRA - repeat$ act as described in the previous subsection, except for building with "make ucq_total_time_test" instead.
- To run $CQRA - double$ and $CQRA - default$ act as described in the previous subsection, except for building with "make UCQ1_MainIncremental" instead.
- To run $MCRA - permute$, refer to the previous subsection.

Important:

(1) The different percentages chosen are hard-coded into UCQ1_Main.cpp in $CQRA - repeat$ and $CQRA - double$.
(2) $CQRA - repeat$'s percentages are listed in the file samplesizes.txt (in *UCQs/UCQ1*) in **absolute** number of answers (instead of percentage). So that must be checked before usage.

Figure 5

To measure time spent on answers .vs. time spent on rejections in $CQRA - default$ and $CQRA - repeat$ for $Q_7^S \cup Q_7^C$ do the following:

For $CQRA - repeat$:

- Go to *UCQs/UCQ1*
- Run "*make ucq_split_time_test"*
- Run the executable three times and average out the results

Note: the different intervals are listed in *UCQs/UCQ1/thresholds.txt* as an absolute number of answers. So it must be checked before using, due to depending on the query size.

For $CQRA - default$:

- Go to *ProbabllisticCounting/UCQ1*
- Run "make UCQ1_MakeTimer"
- Run the executable three times and average out the results

Note: the different intervals for this experiment are listed in percentages in UCQ1_MainTimer.cpp and can be configured easily.