CS386D Term Project Initial Plan

April 27, 2022

1 Team Members

- Long Do
- Sangsu Lee

2 Proposed Starting Point

For the query parser, we plan on using JSQLParser: (https://github.com/JSQLParser/JSqlParser)

3 Proposed Plan

3.1 Problem Statement

In this project, we are interested in measuring two topological features of graph queries in benchmarks called treewidth ($tw(H)$) and hypertree width ($htw(H)$). In particular, we are more interested in measuring the treewidth of both cyclic and acyclic conjunctive queries (CQs).

To measure the treewidth of a acyclic query, we first can convert the query into a hypergraph. From the hypergraph, we can construct a join tree by using the GYO algorithm [3]. We then use the modified version of TB algorithm [1] to measure to treewidth. However, by doing this, we treat cyclic and acyclic CQs as two separated entities. Therefore, we employ the idea of tree decomposition to measure the treewidth of both acyclic and cyclic CQs. Again, we will use the modified version of TB algorithm [1] to do this. Since the algorithm take a graph $G = (V, E)$ as input, we have to construct a primal graph $G = (V, E)$ from the hypergraph $H = (V(H), E(H))$ of the input query.

For part 1 of extra credit, we, again, use the TB algorithm [1] to measure the treewdith $tw(H')$. As mentioned above, the TB algorithm [1] take a graph $G = (V, E)$ as input; therefore, we have to construct we query graph $G = (V, E)$ from the input query.

For part 2 of extra credit, we will employ the $new - det - k - decomp$ algorithm [2] to measure the hypertree width $hw(H)$.

3.2 Algorithm

3.2.1 TB Algorithm

The TB algorithm take a graph $G = (V, E)$ and an upper bound k as inputs and produces a treewidth $tw(H) \leq k$. The default value k in [1] is 12; however, there could be the case that it is impossible to tree decompose a graph $G = (V, E)$ to a tree with $tw(H) < 12$; thus, in this case, we plan to use binary search within the range $[2, |\bigcup_{i=1}^{n} attr(R_i)|]$ to find the optimal value of k.

3.2.2 New Det-k-Decomp Algorithm

Unlike the TB algorithm [1], the $new - det - k - decomp$ algorithm [2] takes s hypergraph $H = (V(H), E(H))$ as input to produce tree decompositions.

4 References

1. Tuukka Korhonen, Jeremias Berg, and Matti Järvisalo. Solving Graph Problems via Potential Maximal Cliques: An Experimental Evaluation of the Bouchitté–Todinca Algorithm. ACM J. Exp. Algorithmics, 24, feb 2019.

2. Georg Gottlob and Marko Samer. A Backtracking-Based Algorithm for Hypertree Decomposition. ACM J. Exp. Algorithmics, 13, feb 2009.

3. Hector Garcia-Molina, Jeffrey D. Ullman, and Jennifer Widom. Database Systems: The Complete Book. Prentice Hall Press, USA, 2 edition, 2008.