Shanghai Jiao Tong University
Lab4 Report: Query Optimization
Name: Yunxuan Xiao
Sid: 516030901268

Database 2019
June 2nd

1 Design Decision

1.1 IntHistogram.java

In this class, I construct a histogram for every attribute in the table. With a fixed number of buckets, and Min Max value, we can calculate the width of each bins. With each bucket representing the number of records in a fixed range of the domain of the attribute of the histogram. The extimateSelectivity() function is quite straight forward, where we compute the AVG, MIN, MAX, COUNT, etc with the constructed Instagram.

1.2 TableStates.java

To initialize a TableStates object, we pass through the table three times. The first scan calculate the min/max value for each attribute. The second scan build up Histograms for each fields. And the last scan put each value in each tuple into the according Histogram.

1.3 Join Optimization

In this section, I implemented the optimizer for join operation following the pseudocode provided on the website.

```
1. j = set of join nodes
2. for (i in 1...|j|):
3.     for s in {all length i subsets of j}
4.       bestPlan = {}
5.       for s' in {all length d-1 subsets of s}
6.            subplan = optjoin(s')
7.            plan = best way to join (s-s') to subplan
8.            if (cost(plan) < cost(bestPlan))
9.               bestPlan = plan
10.      optjoin(s) = bestPlan
11. return optjoin(j)
```

Figura 1: DP optimization

However, when I finish this, I found I can't run the query test in lab3. The simplest selection query will throw an error. It seems that we have to throw a new Vector object if the input *joins* is empty, otherwise it will go run in the query parser. As the unit test is so weak, it's easy to ignore this potential bug.

1.3.1 Still so Slow!

In lab3, it was so slow for my code to finish the second and the third query. I assume that the query optimization will greatly accelerate the joint operation. However, when

I come back to test the queries in lab3, I still cannot finish running in a short time. Finally, I replaces the *join* into *hashequjoin* when the operator is *Equal*. Finally the two runs finished within 5 seconds.

2 API Adaptations

No change in lab4.

3 Difficulty and Time Consuming

I spent around 4 hours in this lab. The most difficulties is to fix unseen bugs across lab1-3. Also the Stack Trace in each broke points in query test is so deep that brings a little problems during coding.