Monte Carlo Tree Search for Generating Interactive Data Analysis Interfaces

Yiru Chen, Eugene Wu Columbia University.

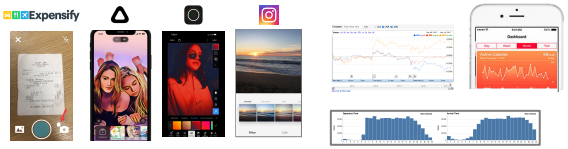
ACM SIGMOD +PODS 2020

Background: Interfaces and Tasks

General purpose interfaces:
Can do a lot, but complex to use.

Excel Photoshop SQL Tableau

Specialized (Precision) interfaces
Designed for specific task, but easy to use, simple



Problem: Building Interfaces is Hard ☹

1. Expensive to learn user's analysis tasks + build interface

2. Dashboard builders limited in analysis complexity or require programming

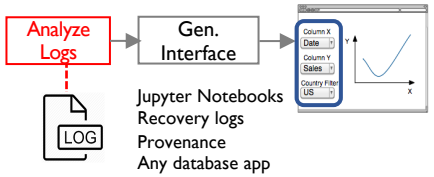
3. Prior work auto-generates forms using database contents, but ignores analysis task may be complex if DB has many tables+attrs

High costs → interfaces mainly built for high profile tasks that are "worth it"

"Unpopular" tasks ignored, but they are important for their users!

Previous Work: Precision interface V1[1]

Query logs as a proxy for analysis tasks to directly generate interface.



Jupyter Notebooks
Recovery logs
Provenance
Any database app

Limitation

1. Visualization & Interaction over Visualization

2. Layout and Usability

Main Contribution

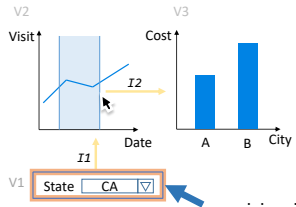
Compared to previous work which only generates a set of widgets, this work enhances the generated interface quality from:

1. Automatically generate an optimal set of **views**, including widgets and visualizations, and the interaction over the views.

2. Consider hierarchical layout as well as the usability in terms of how easy to express the query log.

3. A generic frontend engine which takes a specification as input and output a web app.

What is an Interface?

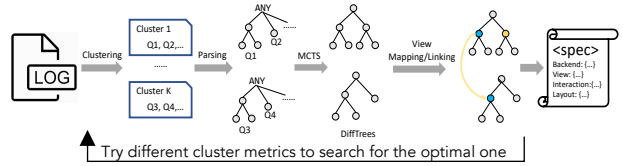


Views: <v1, v2, v3>
Interactions: < I1, I2>
Layout: Horizontal(
Vertical(V2, V1), V3)
Backend: DB

SQL, e.g
Select date, visit from T where state = 'CA'
Select city, cost from R where data between 1 and 10

precision interface V1.

Overview



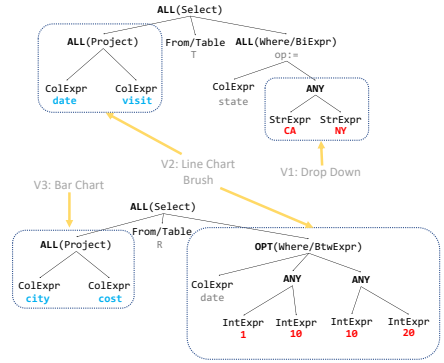
Problem Statement: **Given query log, view types, Find lowest cost interface and layout where all queries can be expressed.**

1. Cluster metrics
   - Project attribute union compatible
   - Overall query similarity etc.
2. Parse: Queries modeled as abstract syntax trees
3. Monte Carlo tree Search:
   - The layout and the selected views are intertwined with the process of identifying subtree differences between the input query ASTs. Search space is very large.
   - Method: UCT algorithm
   - Reward: Randomly assign views and greedily link k times to pick the lowest cost.
4. Cost Estimation for view mapping and linking:
   appropriateness of each view + usefulness + layout score

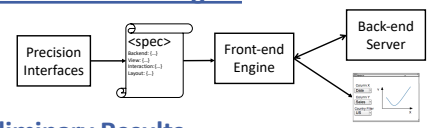
Example: View Mapping and Linking

After we get the final difftree, we search for the optimal view mapping and linking. Layout will be expressed by assigning layout node: Horizontal and Vertical to ALL node.

```
SELECT date, visit FROM T WHERE state='CA';
SELECT date, visit FROM T WHERE state='NY';
SELECT city, cost FROM R WHERE date BETWEEN 1 AND 10;
SELECT city, cost FROM R WHERE date BETWEEN 10 AND 20;
```

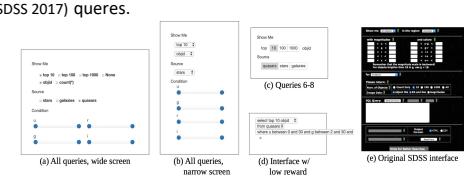


Generic Frontend Engine

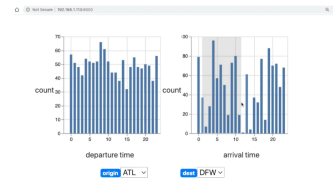


Preliminary Results

Layout sensitive interface generated from 10 Sloan Digital Sky Survey (SDSS 2017) queres.



Interactive visualization interface generated from falcon flights query log.



Acknowledgement: Eugene Wu, Google, NSF, amazon

[1] Zhang, et al. "Mining Precision Interfaces From Query Logs." *SIGMOD*(2019)