MIT 6.830

Applying SimpleDB to d3.js

Research

by

Ermain, Nchinda, Teddy

Abstract

abstract stuff

Contents

1 Introduction

1.1 d3.js

-what is d3

1.2 Motivation

-the class and simpledb -who is holger

2 Library Analysis

We looked through the submodules of d3.js for potential places it could be improved by techniques we learned in 6.830.

Arrays - cross.js, the cross product submodule uses a nested for loop, but to compute cross products which makes sense

Axes - doesn't do anything wilh getting points, just displays data

Chords - don't really think there's much here

Delimiter-Separated Values - only iterates through the elementes of a table

Geographies - this does not use nested loop joins

Hierarchies - this is literally a tri structure, an index on top won't really help

Quadtrees - this is a search library used by other libraries,

Scales- does not work with combining points, uses interpolaters to translate an input into value in a range

Selections - there are nested for loops everywhere

Time Intervals - there are no nested loops here, but maybe for large intervals it would be better to get $O(logn)$ time queries instead of $O(n)$. This isn't our main focus though

Zooming - no nested loops found, likely offloads bulk of processing to quadtrees -Forces -based on quadtrees, can't really do anything here -Transitions - https://github.com/d3/d3-transition/blob/master/src/transition/se we could do something here

3 Improvements

-make a boolean flag to enable index creation -create an index based on user provided id for nodes and comparator, when new nodes are added make sure to add them to the index too

4 Results

-how does our text example run with this new code

5 Conclusion

6 Acknowledgements

References

[1] https://github.com/ermain/d3js_experiments

[2] https://github.com/d3/d3/blob/master/API.md