Visualization of Semantic Windows with SciDB Integration

Hasan Tuna Icingir
Department of Computer Science
Brown University
Providence, RI 02912
hti@cs.brown.edu

February 6, 2013

Abstract

Interactive Data Exploration Using Semantic Windows[1] offers a solution to finding interesting patterns and objects from a big set of data where the users do not need to wait for the whole query to finish and have the ability to see intermediate results or receive progress updates. This project includes the visualization component of Interactive Data Exploration Using Semantic Windows. Since the Interactive Data Exploration Using Semantic Windows requires that the database should be able to execute range queries efficiently, another aspect of this project is its integration to SciDB[2] which is especially optimized for the management of big data.

1 Introduction

Most database management systems lack interactivity. When a user runs a query, he has to wait for the whole query to finish to get an actual result and this might take a very long time. Interactive Data Exploration Using Semantic Windows can display intermediate results in queries and the addition of a visual component to this system will make it even more user-friendly. The visualization component of this project enables the users to observe the results of a query via live graphs.

Interactive Data Exploration Using Semantic Windows uses PostgreSQL as its back-end database management system and the other aspect of this project was to integrate it to SciDB. SciDB is specifically designed to solve data-intensive problems, so it's a good choice for the back-end database system. Since the data files are huge and most queries take a long time to execute, SciDB integration was significant and actually improved the overall running times of queries.

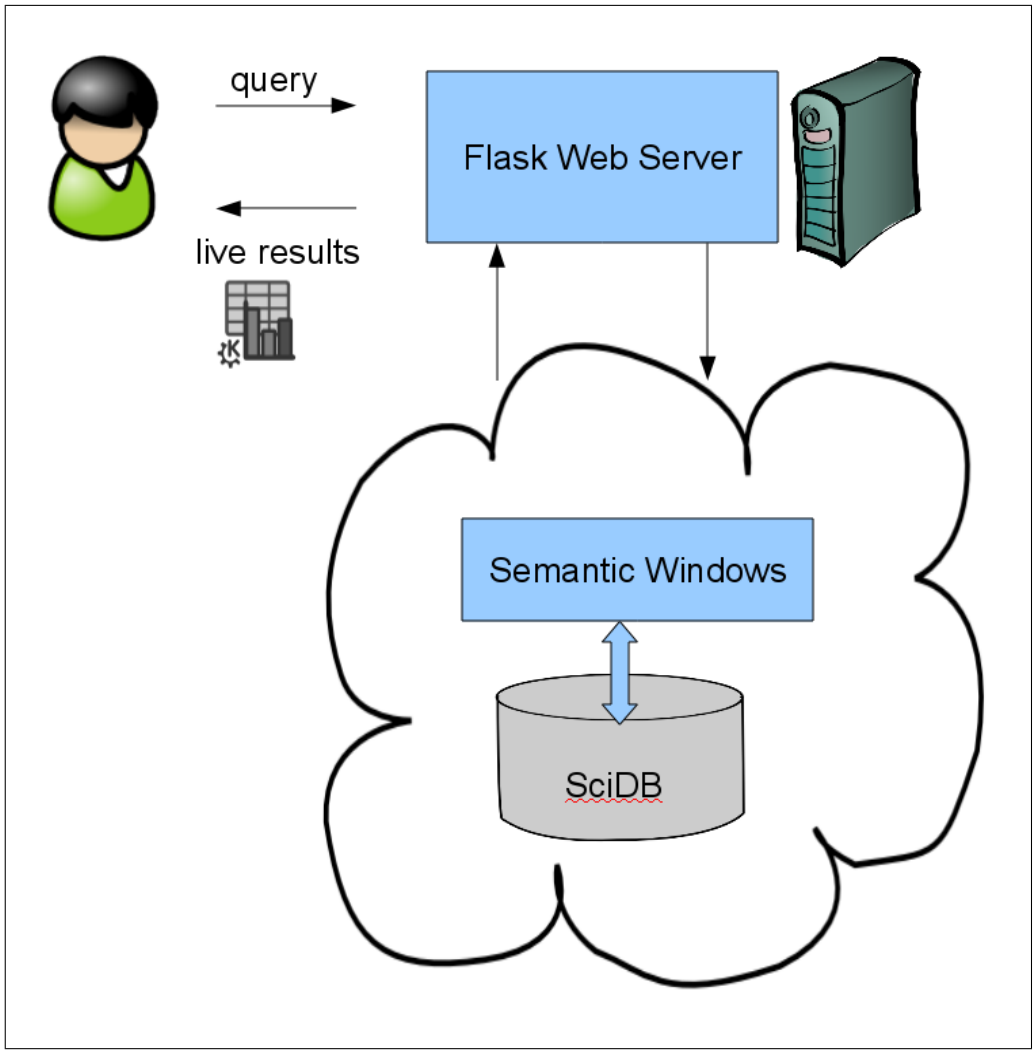

Figure 1: System Architecture

2 System Architecture

The system that was built for this project integrates each of the following list of components and technologies:

- Back-end DB : SciDB
- Web server : Flask Python Framework[3]
- Visualization Framework : D3[4]
- Scripting language : Python[5]
- Front-end : HTML/Javascript
- Query representation : XML

3 An Example Query and Its Visualization

The windows that we are referring to in Interactive Data Exploration Using Semantic Windows are the regions where the user is querying for. So, the regions will be displayed as they are found without waiting for the whole query to finish. While creating the visualization part of this system, we worked on two data sets: SDSS[6] and synthetic data. Sloan Digital Sky Survey (SDSS) data is formed after the observation of sky and both data-sets are huge, so they are of good use for the problem that we are trying to solve.

An example query that one can run on this data set would be finding regions where the sky's average brightness is more than x. For this query, the visualization component displays the output on an XY-plot as the regions are found. So, the coordinates that fulfill the query are displayed on a live graph and the user does not have to wait for the whole query to finish. The details and the characteristics of the visualization tool will be explained in further detail.

4 SciDB Integration

In order to use SciDB as the back-end database, we had to convert all the data sets to the format that SciDB uses. SciDB utilizes arrays to hold the data and we had to convert our data to this special format. The queries were being run on two big data sets named SDSS and synthetic data and both of these data sets were previously in the CSV format. SciDB has a command to convert .csv files to .scidb named csv2scidb.

After converting the data to the file type that SciDB supports, we created the arrays and loaded all the data to them. Mind that the arrays we used are two dimensional and they contain gigabytes of data about the sky. So, the example in the previous section about finding specific regions from the sky can be queried on SciDB after we processed the data and loaded it to SciDB arrays. The sample SDSS array that we formed is represented like this:

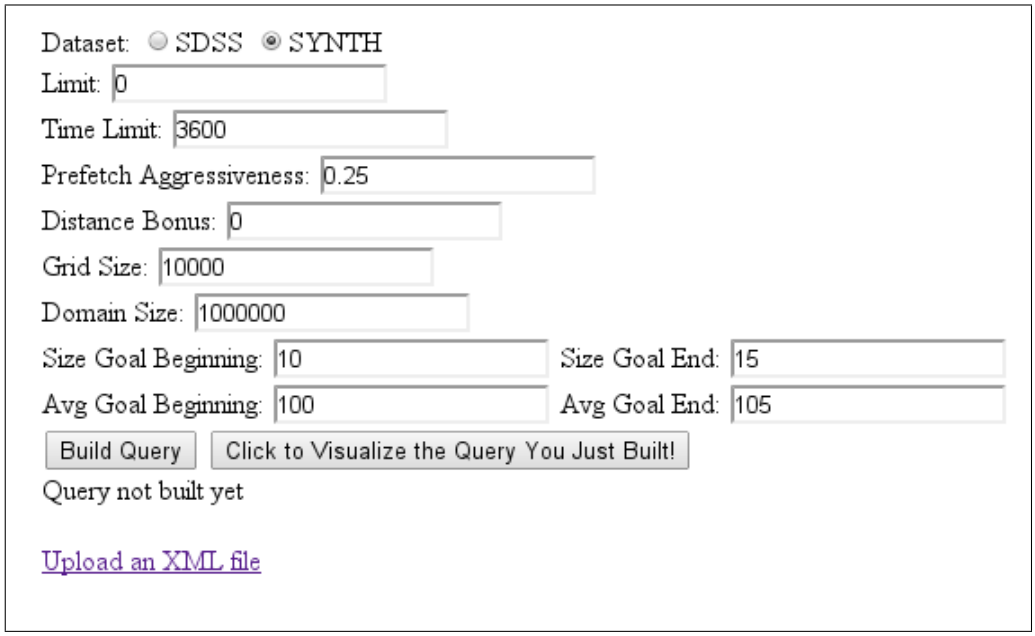

Figure 2: Building the Queries Online

```
create array sdss_sample_2d
  <raerr:double,
  decerr:double,
  objid:int64,
  skyversion:int16,
  run:int16,
  rerun:int16,
  mode:int16,
  type:int16,
  rowv:double,
  colv:double,
  rowverr:double,
  colverr:double>
    [ra=0:*,500,0,
    dec=0:*,500,0];
```

The Semantic Windows framework executes the SciDB queries and outputs all the intermediate results. The results are then parsed by the web server and displayed on the browser on a live graph which is built with D3.

5 The Web Server

We wanted the users to access the results online via their favorite browsers; therefore, a web server was necessary to build this system. I used the Flask framework to create this server since it's very convenient and since it's run as a Python application. The web pages are built with HTML/Javascript and the graphs are plotted with the D3 framework, which can all easily be rendered by Flask.

The web server runs on a department machine which has access to SciDB and the Semantic Windows framework; therefore, the queries can be passed on to Semantic Windows which uses SciDB as it's back-end database. The first thing we want from the user is the query, which is represented in an XML format. There are two options for the user to supply the query to us from the main web page. She can either upload the XML file from her hard disk or use the HTML form on the web page. Uploading the query files is straightforward but the user might not have any experience with our query style; so, we parse the inputs from the web page and create an XML file using the lxml[7] library of Python.

After we receive the query from the user, either from an uploaded file or from a form, we pass it to the Semantic Windows application which will send us the results as they are found. These results need to be visualized in a user-friendly way and that's where D3 is used.

6 Visualization of the Output via D3

As explained in the example query part of the report, one can use Semantic Windows to find specific regions of the sky. When this query is run and we start getting the output, the output is parsed in the Python application of the web server and is then sent to the visualization web page which uses D3 to display the graphics. We know that we need to plot the areas on an XY-graph, so after extracting the grid sizes and the domain from the query's XML file, we first plot the empty graph and then wait for the coordinates to come from the server.

As the coordinates arrive from Semantic Windows, they are plotted on the graph as shown in figure 3. This is a live graph, meaning that it displays the data as they arrive and gives the user the chance for interaction. The query that is being run is displayed at the bottom of the page and when the user hovers her mouse on the plotted shape, the shape that is being observed changes its color and information about that shape is displayed below.

D3's graphing tools are used during the depiction of these data. The rectangles are represented with the coordinates that are received from Semantic Windows and the XY-graph is built with the grid and domain values from the query file.

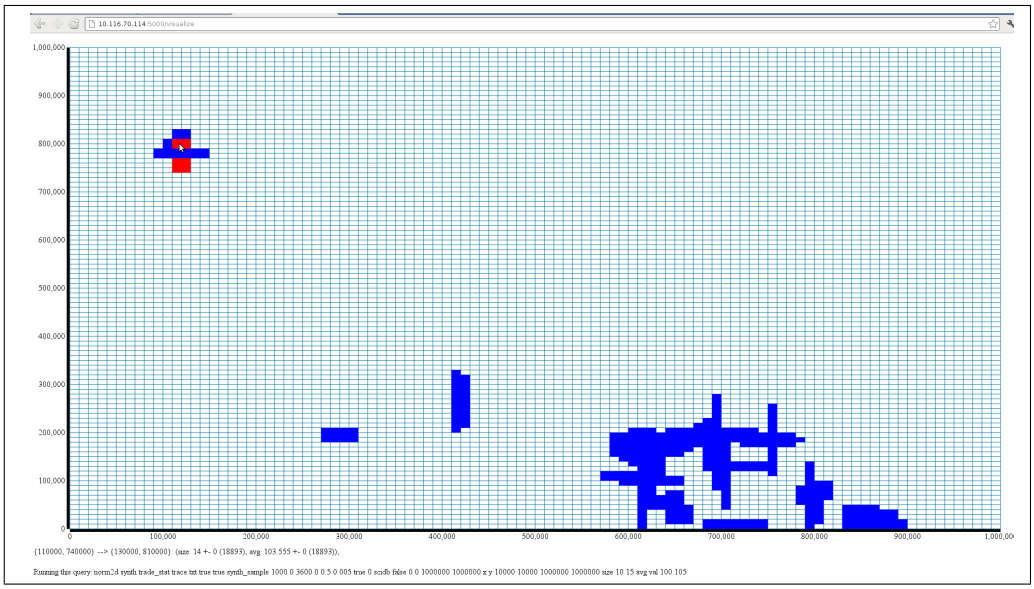

Figure 3: Visualization of the Output

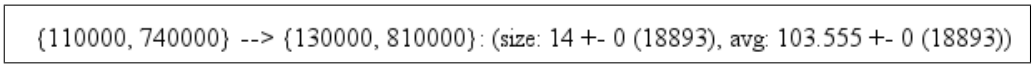

Figure 4: Information is Displayed With Mouse Hovering

7 User's Manual

In order to use this visualization component with Semantic Windows on SciDB, you need to:

- Install and configure SciDB on your machine,
- Install Semantic Windows framework,
- Install the Flask framework,
- Run the Python application that starts the web server (has to be in the same folder as Semantic Windows),
- Locate visualization.html which uses D3 and index.html under a folder named templates,
- Query files in XML (optional),
- Connect to local host while the web server is running and run the queries from your favorite browser,
- Click Visualize and observe the results you get from the live graph by hovering your mouse on the shapes.

8 Future Work

This project is open to further improvements according to the needs of the users and the ways of interactivity they wish to have with their query results.

- The query building page can be improved by asking for more input and the user's can set their own goals instead of filling out the template queries.
- The shapes can be more informative and they can display the queries in SQL, AQL and AFL during user interactions.
- Other types of graphs can be supported and the shapes can be displayed in a more informative way (lighter or darker).

Acknowledgments

I would like to thank Alex Kalinin, the main author of Interactive Data Exploration Using Semantic Windows, and Ugur Cetintemel who gave me lots of significant suggestions throughout the advancement of my project.

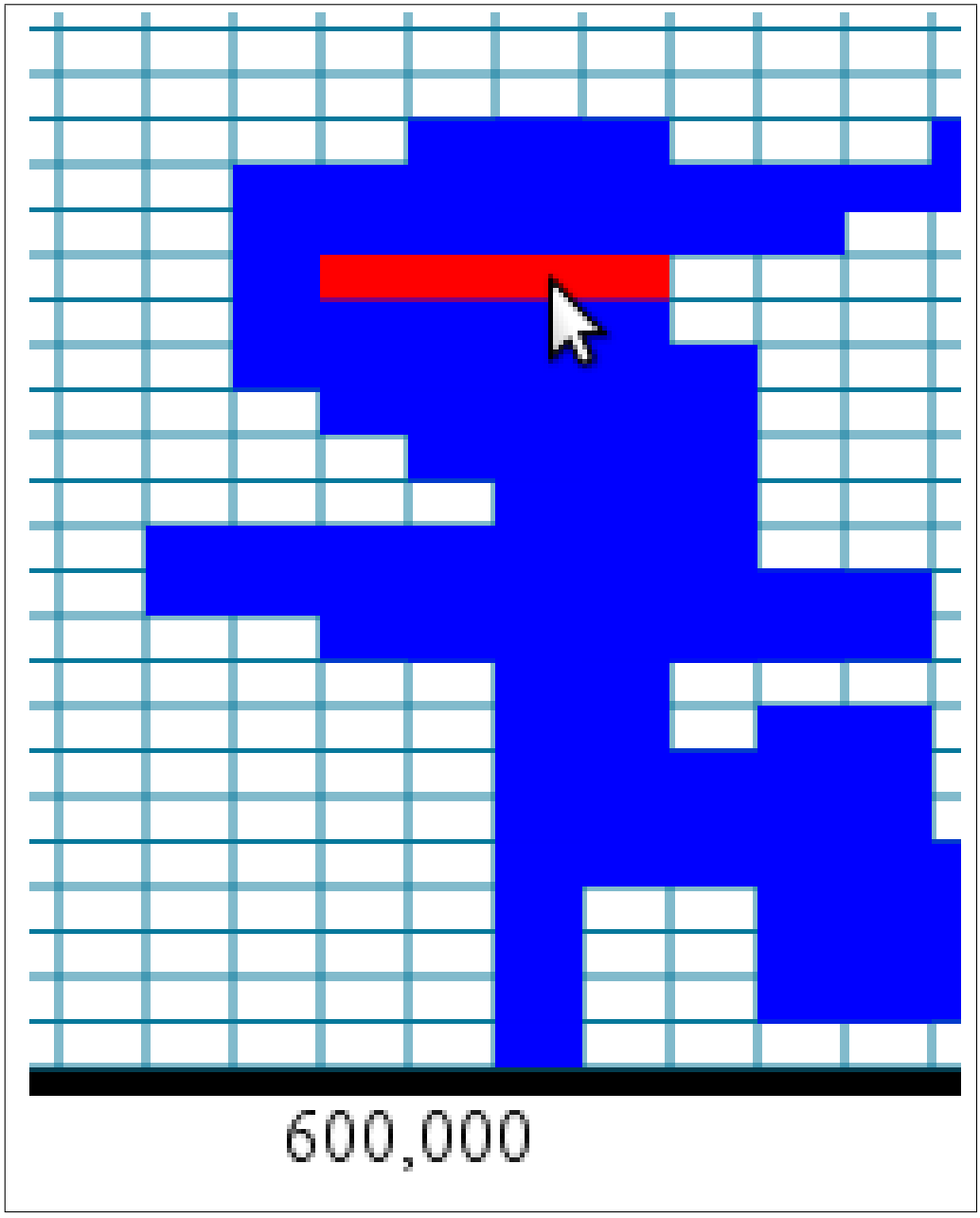

Figure 5: Zoomed View of a Cluster

References

[1] Alex Kalinin, Ugur Cetintemel, Stan Zdonik. Interactive Data Exploration Using Semantic Windows. (not published yet)

[2] SciDB `http://www.scidb.org/`.

[3] Flask Microframework for Python `http://flask.pocoo.org/`.

[4] Data-Driven Documents. `http://d3js.org/`.

[5] Python Programming Language `http://www.python.org/`.

[6] The Sloan Digital Sky Survey `http://www.sdss.org/`.

[7] lxml - HTML and XML for Python `http://lxml.de/`.