CIS 600: MASTER’S PROJECT

PROJECT REPORT ON

“RESEARCH AND ANALYSIS OF MATRIX OVERLAPING AND REPRESENTATION WITH JAVASCRIPT “

Presented By:
Virbhadrasinh Kumpavat
Student Id: 01592422

Under the Guidance of:
Prof. Maoyuan Sun
Department of Computer and Info Science

May-2017

**DEPARTMENT OF COMPUTER & INFORMATION SCIENCE
UNIVERSITY OF MASSACHUSETTS -
DARTMOUTH**

Table of Content

Chapter 1: Introduction

1.1 Introduction to the project

The aim of the project is to classify the multiple matrix similarities based on overlapping part and visualize it with JavaScript with d3.js. The overlapping of the matrix is representing the similarities between two or more matrix with overlapping part. In this project we develop user friendly visualization for the matrix overlapping by use of the clustering in JavaScript, create matrix with group by element with d3.js library and try to show similarities based on area that split with each other. With use of visualization user can find out the maximum possibilities of overlapping between two matrix with use of common Numbers of rows and columns. Here User can select the numbers of Rows and Columns and system generate the two matrixes and calculate the difference of Rows and Columns with Mathematic equation and show the maximum possible overlapping part between this two User created matrixes.

1.2 Technical Specification

The programming Code has been written in Text Editor. We use SVG and D3.js Library to create visualization with use of JavaScript .To provide user interface we used HTML and CSS3.

JavaScript:

JavaScript is an object-oriented computer programming language commonly used to create interactive effects within web browsers. JavaScript often abbreviated as "JS", is a high-level, dynamic, untyped, and interpreted run-time language.[1] It has been standardized in the ECMAScript language specification. Alongside HTML and CSS, JavaScript is one of the three core technologies of World Wide Web content production; the majority of websites employ it, and all modern Web browsers support it without the need for plug-ins.

JavaScript is prototype-based with first-class functions, making it a multi-paradigm language, supporting object-oriented. Imperative, and functional programming styles. It has an API for working with text, arrays, dates and regular expressions, but does not include any I/O, such as networking, storage, or graphics facilities, relying for these upon the host environment in which it is embedded.

SVG: Scalable Vector Graphics

Scalable Vector Graphics (SVG) is an XML-based markup language for describing two-dimensional vector graphics. SVG is essentially to graphics what HTML is to text.SVG is developed by W3C ("World Wide Web Consortium") in September 2001[2]

SVG is similar to Adobe's proprietary Flash technology, but it is a text-based open Web standard instead of a closed binary format. It is explicitly designed to work with other web standards such as CSS, DOM, and SMIL. SVG drawings can be dynamic and interactive.

D3.js – (Data Driven Document)

D3.js written in JavaScript and Developed first by "Mike Bostock" with his team in Feb-2011 at Stanford University. [3]

D3.js is a JavaScript library for manipulating documents based on data. **D3** helps you bring data to life using HTML, SVG, and CSS. D3's emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation. [4]

D3.js allows you to bind arbitrary data to a Document Object Model (DOM), and then apply data-driven transformations to the document. For example, you can use D3 to generate an HTML table from an array of numbers. Or, use the same data to create an interactive SVG bar chart with smooth transitions and interaction.

D3 is not a monolithic framework that seeks to provide every conceivable feature. Instead, D3 solves the crux of the problem: efficient manipulation of documents based on data. This avoids proprietary representation and affords extraordinary flexibility, exposing the full capabilities of web standards such as HTML, SVG, and CSS. With minimal overhead.D3 is extremely fast, supporting large datasets and dynamic behaviours for interaction and animation. D3's functional style allows code reuse through a diverse collection of components and plugins.

D3.js is used on hundreds of thousands of websites. Some popular uses include creating interactive graphics for online news websites, information dashboards for viewing data, and producing maps from GIS map making data. In addition, the exportable nature of SVG enables graphics created by D3 to be used in print publications.

The other Requirements are

1. Operating System - Windows 7 or above

2. RAM- 4GB

3. Disk Space – 4GB

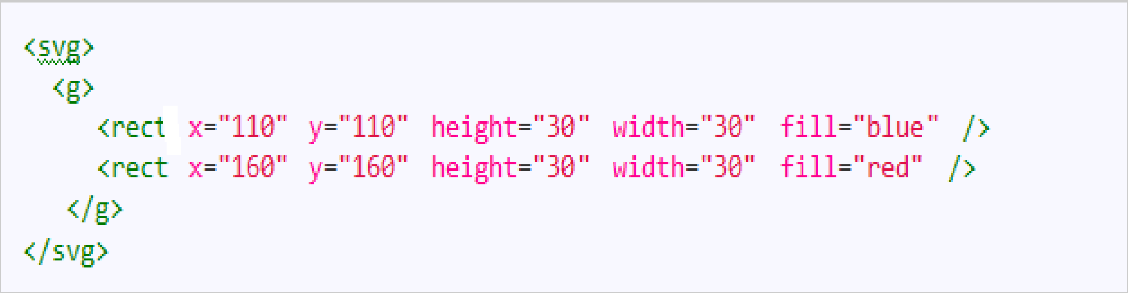
Chapter 2: Project Description

This project is done in mainly four part Analysis, Create matrix with user input, Calculation of Overlapping part and Presentation the maximum overlapping.

During the Analysis about the matrix representation there is no any visual presentation or any example about the matrix visualization also we could not find any tools to represent the matrix and maximum overlapping of matrix. So we are creating matrix visualization for the maximum overlapping.

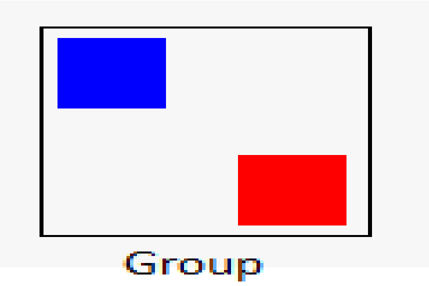
To create an adjacency matrix First we create one cell (Normal Square). For more Rows and Columns we cluster that multiple cell in one group (Group by element) by use of SVG group element present in Rows or columns. In this project user gives input for the number of rows and number of columns As per user input the algorithm create a multiple cell and combine into group (<g> element) and Shows matrix.

Example of Grouping: In this example we just group two small rectangles in one group with use of SVG group elements.[5] both rectangles have different coordinates.



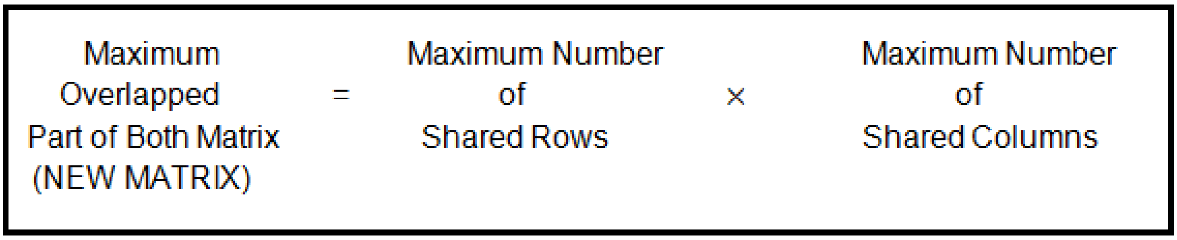
```
<svg>
  <g>
     <rect x="110" y="110" height="30" width="30" fill="blue" />
     <rect x="160" y="160" height="30" width="30" fill="red" />
   </g>
</svg>
```

SVG Group Element



Calculation of overlapping is third part. In this part after getting the values for Number of Rows and Numbers of columns for both matrix. System Create both Matrix1 (RED) and Matrix2 (BLUE) based on user input. And also calculate shared maximum possible number of Rows and Number of Columns.

There is mathematic Equation to find out maximum overlapping part, to show maximum overlapping part we need maximum shared rows and Maximum shared columns.



$$\text{Maximum Overlapped Part of Both Matrix (NEW MATRIX)} = \text{Maximum Number of Shared Rows} \times \text{Maximum Number of Shared Columns}$$

Figure3: Equation to find out maximum possible overlap part

So, Compare Number of Rows and Numbers of Column of Matrix 1 and Matrix 2 we can get Number of Rows and Number of Column (contributions) for Matrix3 (cover part). The base Rows and Columns From both grid are contribution for cover part. With utilization of contribution for new lattice the framework produce the Maximum covering part and show at base in the framework. The Green matrix is maximum overlapped part from both matrix.

Here in visualizations we are provide functionality where user can create matrix based on requirement. After creating the matrix we use DRAG function of D3.js library so user can drag each matrix on the screen or make any position of overlapped and figure it out how many possibilities of overlapping are available for those two matrix and based on this two matrix our system generate the maximum overlapped part of this two matrix.

Chapter 3: Design and Implementation

For this project all code done with JavaScript and d3.js library. To creating matrix overlapping visualization first we need matrix and there are no particular tools or any reference which provide matrix in JavaScript. So we decide to create own matrix design frame work where user can create a matrix based on requirement. To create whole design first we started from the cell. Than cluster this cells to in group to make matrix

Step1: Create a cell.

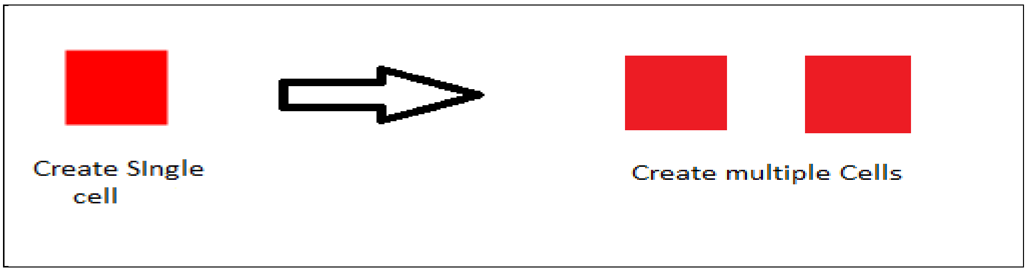
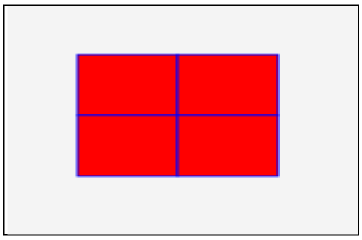


Figure4: Single cell and multiple cell

Step2: Group together

We combine all cells into the one group to make rows and columns. And place next to eah other so they work as a group and make a matrix. Here an example of 2 rows and 2 column (4 small cells) we create with all cells in one group it's create 2*2 matrix.



Group of 4 small cell (Small matrix 2*2)

Step 3: Create two matrixes

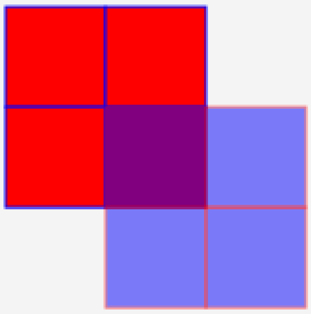
After creating one small matrix we create another one matrix to find out the possibilities of overlapping.

Group of 4 small cell (Small matrix 2*2)
Group of 4 small cell (Small matrix 2*2)

Step 4: Manually find out the possibilities of overlapping.

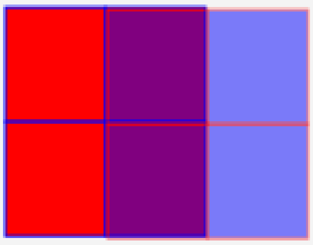
In this stage we just physically cover this two frameworks and discover the potential outcomes of covering, we put the matrix on another matrix. For this situation we simply indicate 25% and half cover of matrix since this two are the most extreme conceivable outcomes for overlapping.

<u>25 % overlap:</u>



25 % overlap in two 2*2 matrix

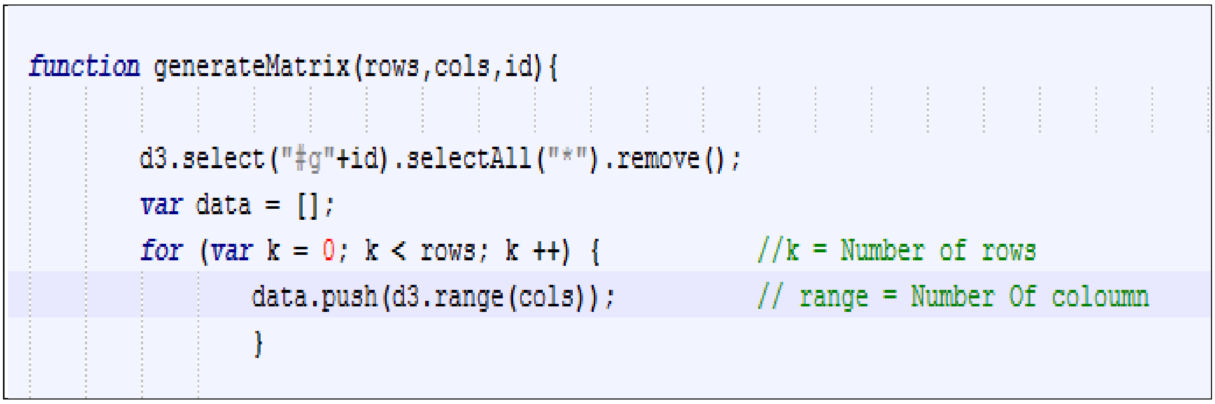
<u>50 % overlap:</u>



50 % overlap for two 2*2 matrix

This 50% overlap is also the maximum overlap .we can easily find out maximum possible overlap here because these two matrixes are small. For big and different size of matrixes we need to calculate the maximum possible overlap part so for that we create user interactive system to find out maximum possible overlapping of two matrices.

To make simple and effective this overlapping system we use user interaction and automation For Automation here we implement one algorithm[6]. The system takes inputs from user in forms of Number of Rows and Number of columns and generates the matrix based on that inputs.



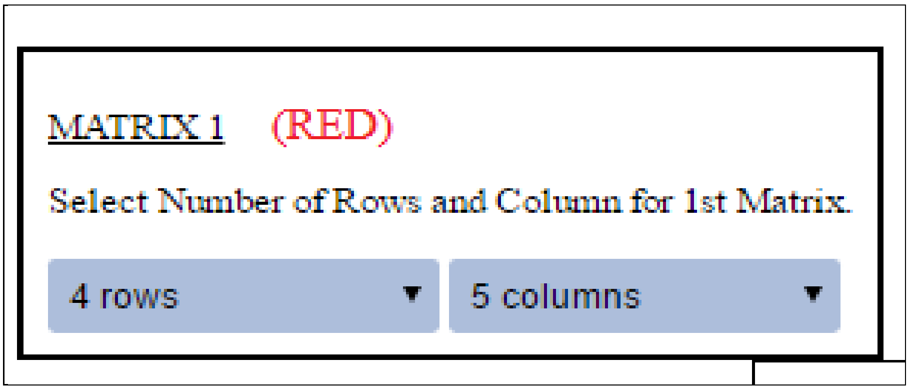
```
function generateMatrix(rows,cols,id){

        d3.select("#g"+id).selectAll("*").remove();
        var data = [];
        for (var k = 0; k < rows; k ++) {          //k = Number of rows
                data.push(d3.range(cols));          // range = Number Of coloumn
                }
```

Figure 7: Function to create matrix

In this function K and Range’s value changed based on user input in the system. Value of K generate the Numbers of Rows and value of Range generate Number Of columns for matrix.[6][7] User input the number of rows and columns as per requirement.

This Selection system generate the matrix after the function run. To create both matrix

Matrix 1:

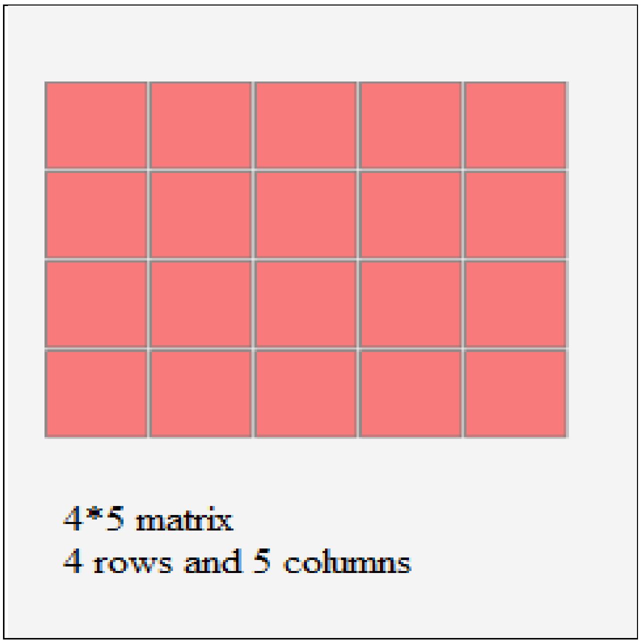


MATRIX 1 (RED)

Select Number of Rows and Column for 1st Matrix.

Figure 8: Input for Matrix 1

Take a look at example. If user need 4*5 matrix , 4 rows and 5 columns, User need to put value in selection box of matrix generator after the selection of 4 rows and 5 columns the values changed for K and Range and function work with value 4 and 5 and generate matrix

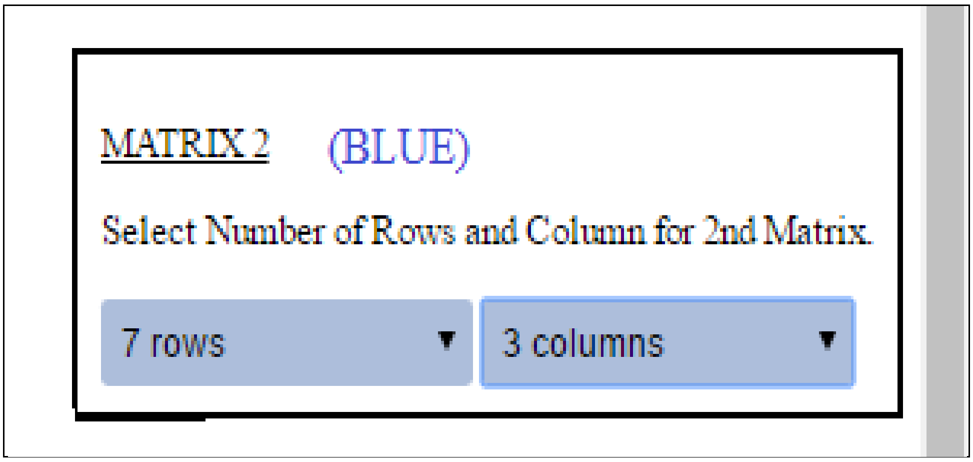


4*5 matrix
4 rows and 5 columns

Figure 9: Matrix 1 based on user input

Matrix 2:

Now we are creating matrix 2 with the user input same as matrix 1. This time if user change the inputs for new matrix. System generate matrix on new inputs.



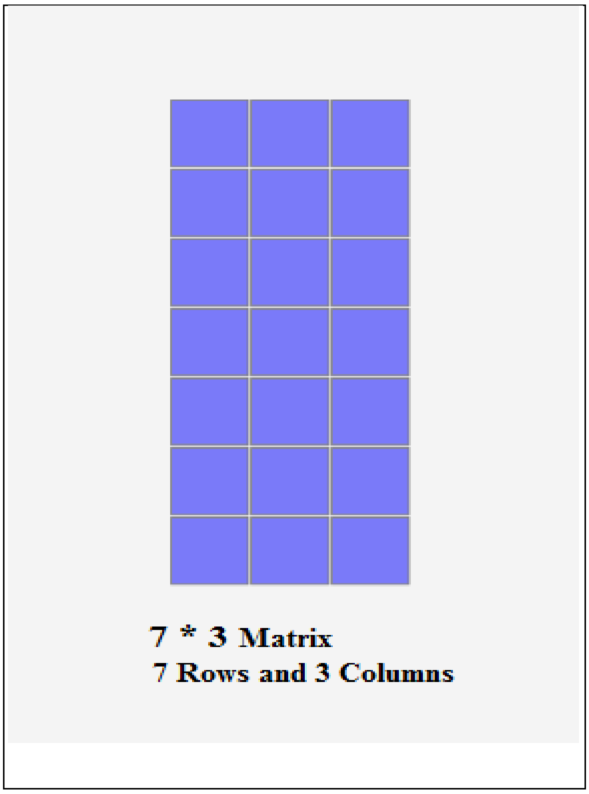
MATRIX 2 (BLUE)

Select Number of Rows and Column for 2nd Matrix.

7 rows ▼ 3 columns ▼

Figure 10: Input for Matrix 2

For matrix 2 user give input for 7 rows and 3 columns so the value of K and Range changed according to input so system take input and generate matrix for 7*3 . So the output from the system is shown as BLUE matrix.



7 * 3 Matrix
7 Rows and 3 Columns

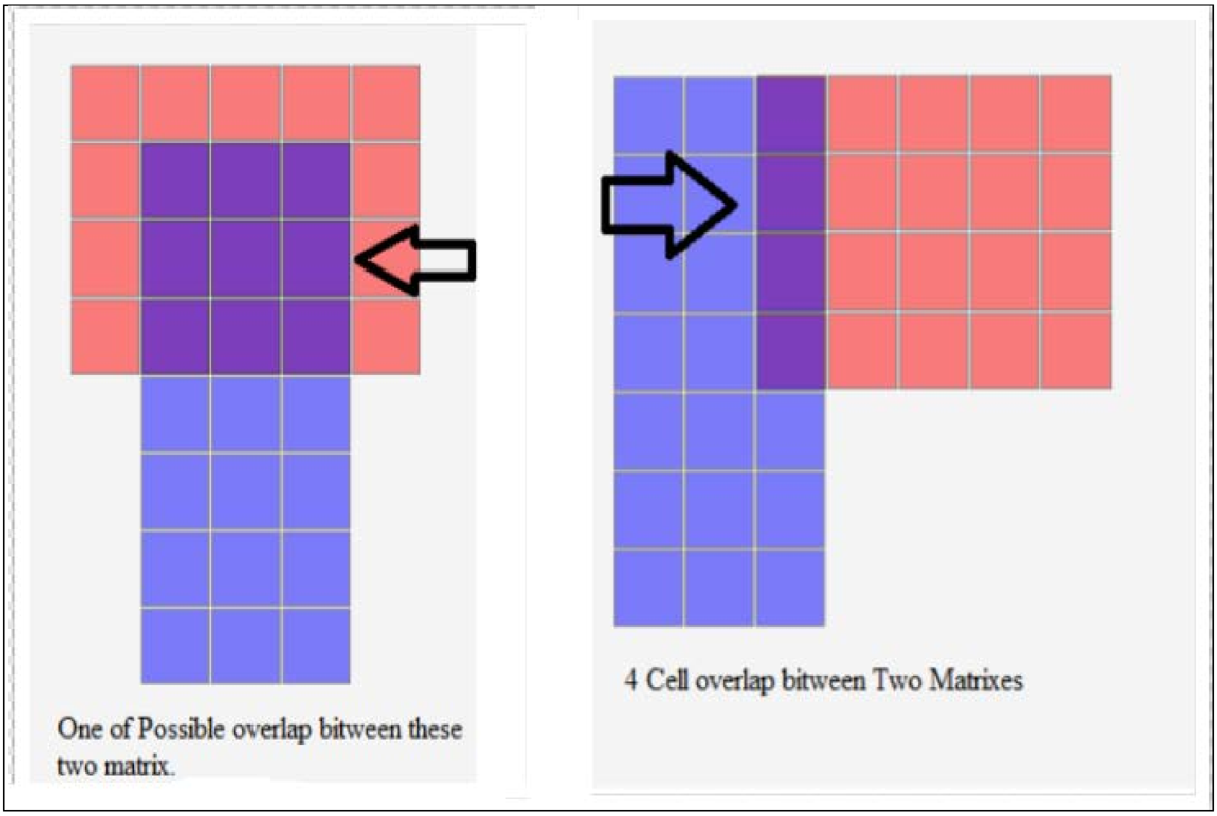
Figure 11: Matrix 2 based on user input

After getting both matrixes now if user want to find possible overlap between this two Matrixes so there are many possibilities for overlap between these two matrixes. To find out possibilities between these two matrixes we add new functionality here its DRAG.

DRAG: .call (drag);

Drag is function of D3.js library with use of this function we can drag the single object or whole group in to the SVG.[8] So here With use of Drag Function user can drag whole matrix to over second matrix with mouse . User just click on matrix and drag on other matrix and try to find out maximum possibilities of overlap.

Here is Some Examples of possible overlap of these two matrixes.



One of Possible overlap bitween these two matrix.

4 Cell overlap bitween Two Matrixes

Figure 12: Shows two possible overlap between two matrixes.

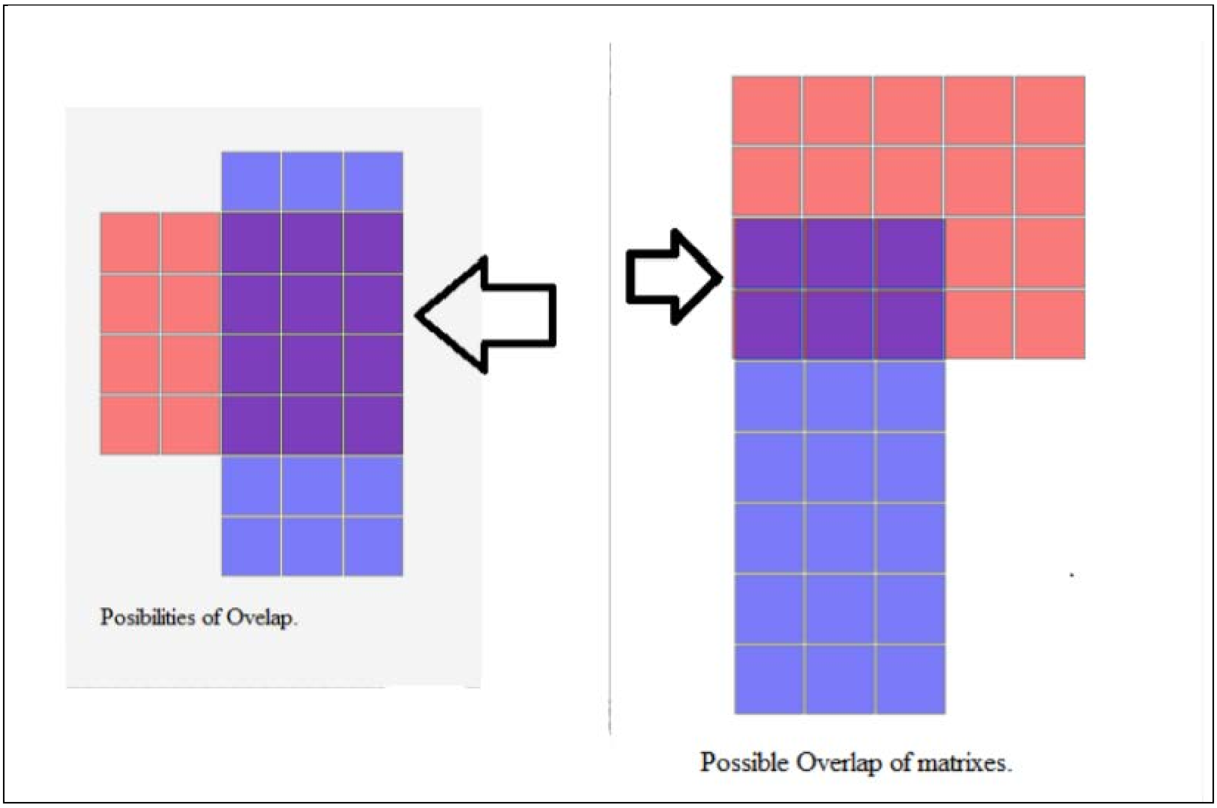


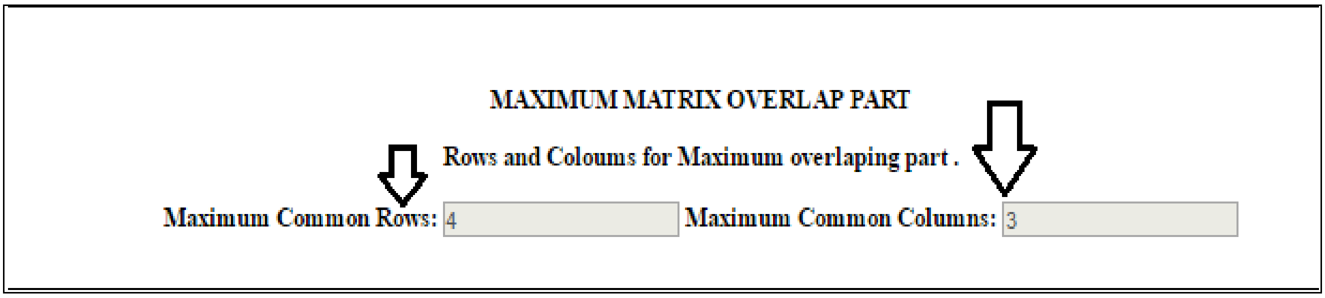
Figure 13: Shows two possible overlap between two matrixes.

These to matrixes are overlap each other to many ways but here we are try to find out maximum overlap possible is between these two matrixes .so we calculate the rows and columns of these two matrixes and find out the Rows and Columns For maximum overlap part . As per equation we seen before get the Number of rows and Columns from Both matrixes.

Here from our example

Matrix 1: 4 Rows * 5 Columns
Matrix 2: 7 Rows * 3 Column

After the Calculation we get Rows and Columns for Maximum Overlap is Most Common Number of Rows and Columns:



MAXIMUM MATRIX OVERLAP PART

Rows and Coloums for Maximum overlaping part .

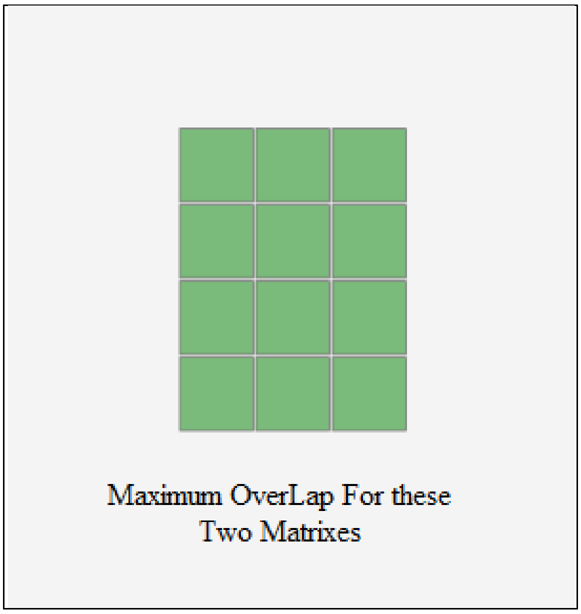
Maximum Common Rows: 4 Maximum Common Columns: 3

Figure 14 : Inputs for Maximum Matrix overlaop

After Get maximum common Rows and Maximum Common Columns the generate Matrix function generate new matrix and this new matrix is maximum possible overlapped part between these two selected matrix. So here,

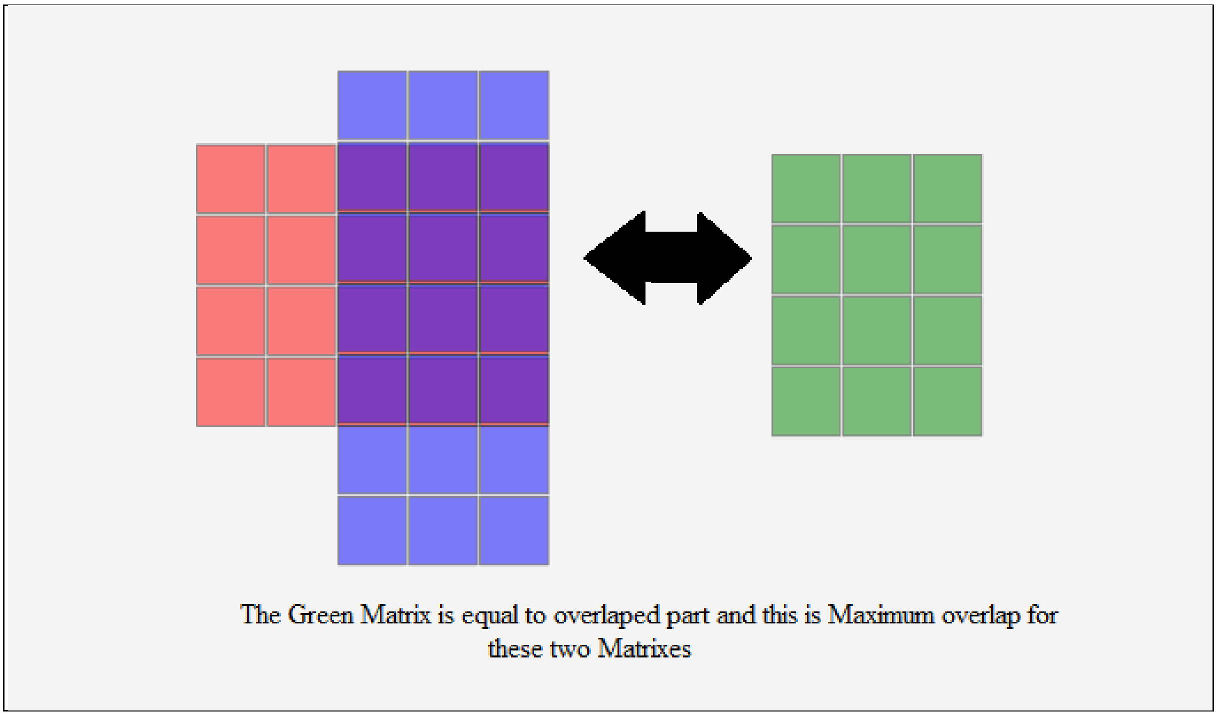
For Matrix 1: 4 Rows * 5 Columns
Matrix 2: 7 Rows * 3 Columns.

Maximum overlap part is 4 Rows * 3 Columns.



Maximum OverLap For these Two Matrixes

Figure 15: Maximum Overlap



The Green Matrix is equal to overlaped part and this is Maximum overlap for these two Matrixes

Figure 16: Both matrixes with maximum overlapped part

DRAG and DRAG OFF Buttons:

Drag and Drag Off buttons give usefulness of drag network into the SVG. At the point when client Click on DRAG client ready to drag Matrixes in entire SVG. On the off chance that client would prefer not to drag those Matrixes into the SVG than DRAG OFF catch is the arrangement.

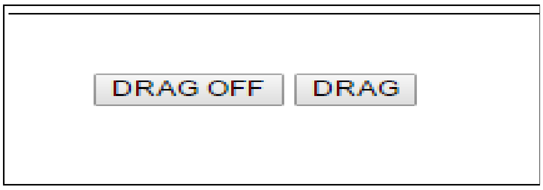
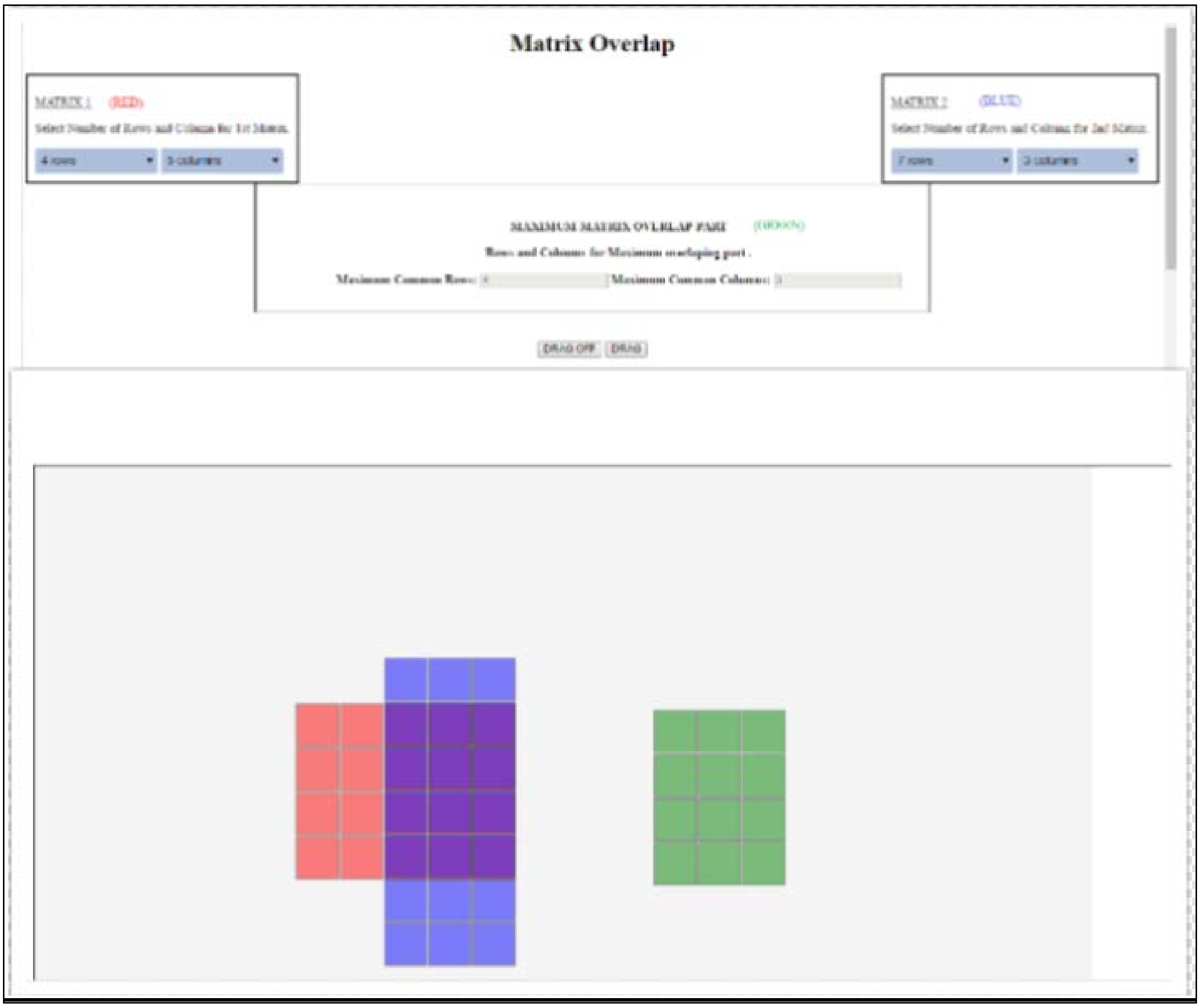


Figure 17: Darg And Drag Off buttons

The Final View:



Matrix Overlap

MATRIX 1 (RED)

Select Number of Rows and Columns for 1st Matrix.

4 rows ▾ 5 columns ▾

MATRIX 2 (BLUE)

Select Number of Rows and Columns for 2nd Matrix.

7 rows ▾ 3 columns ▾

MAXIMUM MATRIX OVERLAP PART (GREEN)

Rows and Columns for Maximum overlaping part .

Maximum Common Rows: 4 Maximum Common Columns: 3

Figure 18: The final Design Of the System

Chapter 4: Testing

In testing we try much possible combination of Number of rows and Number of columns for both matrixes and try to get different result. All time we get correct results for matrix overlapping this system is well tested and provide correct information about overlapping, after testing ads some CSS to buttons and try to make attractive for user. We add buttons for DRAG and DRAG OFF .these buttons also work properly and when user press DRAG OFF the drag function off its work.

Chapter 5 : Conclusion and Future Scope

In this project, we developed Matrixes with use of JavaScript libraries. With this system we can build multiple matrix representation compare them and try to find similarities of those matrixes and overlapping part show maximum possible similarities .We can find out the maximum possible overlapping. So now we have clustering algorithm so in future we can make maximum sized matrix with this algorithm also its very useful to compare other matrixes .In Future we can combine some data sets with this matrixes and make responsive and user interactive visualization with this matrix representation.[9] This Visualization is easy to explain and give brief view about the dataset and very useful for compare dataset and figure it out common things from data sets.

Chapter 6 References

1. https://en.wikipedia.org/wiki/JavaScript
2. https://developer.mozilla.org/en-US/docs/Web/SVG
3. https://github.com/d3/d3/wiki
4. https://en.wikipedia.org/wiki/D3.js
5. https://www.dashingd3js.com/svg-group-element-and-d3js
6. Stackoverflow.com
7. Stackoverflow.com + research on web
8. http://bl.ocks.org/biovisualize/1197731
9. https://www.sas.com/en_us/insights/big-data/data-visualization.html
10. http://ieeexplore.ieee.org/document/1509099/
11. https://github.com/d3/d3/wiki/Gallery
12. http://bl.ocks.org/srosenthal/2770072