Asemap

- A Paper Searching Engine -

Project Report

Group 7

Shanghai Jiaotong University
IEEE

电子信息与电气工程学院
School of Electronic, Information and Electrical Engineering

Title:
Asemap

Theme:
Paper Searching engine

Project Period:
Summer Semester 2017

Project Group:
7

Participant(s):
Jiayu Xu
Ruofeng Liu
Zekun Jiang
Jiaqi Ma

Supervisor(s):
None

Copies: 1

Page Numbers: ??

Date of Completion:
June 24, 2018

Abstract:

A project that is aimed to train the ability to deal with database and construct our own web.

Chapter 1

Front-end

The front end of a website is one of the most important element of the whole project, because the aesthetic measure of the web determinant the popularity of your pages.

1.1 Introduction

To make a beautiful page, you need to have three tools, they're **Html, Css, Javascript**. And among these **Html** is responsible for the content of your page, including words, images, media and so on. **Css** can be ragarded as the decorations of the presentation of the **Html**, it's the core of your web design. As for **Javascript**, it can help you make some dynamic effect, which can make your pages more lively.

1.2 Preparation

To be honest, NOTHING. But you could also use **Sublime** or other text editor to make it.

1.3 Usage

Because the **Html** is so basic that, here, i don't introduce more.

1.3.1 CSS

```
    <style>
        .welcome{
            padding-top: 20%;
            padding-bottom: 20%;
            margin:0;
```

```
            background: black;
            color:white;
            font-family: Arial Rounded MT;
            font-weight: bold;
            font-size:150%;
            cursor: default;
            text-align: center;
            width: 100%;
            height: 100%;
        }
        .search{
            background-color: #3366ff;
            color:white;
            border:none;
            height:31px;
            width:80px;
            cursor:pointer;
            transition: all 0.3s;
            float: left;
        }
        .search:hover{
            background-color:white;
        }
        .search:visited{
            background-color: #3366ff;
            color:white;
            border:none;
        }
        .search:active{
            background-color: #3366ff;
            color:white;
            border:none;
        }
        .module{
            padding: 0;
            margin:0;
            height: 100%;
            font-family: Microsoft YaHei;
            font-weight: bold;
            color:white;
            font-size:150%;
            cursor: default;
        }
        .module ul{
            letter-spacing: -5px;
            height: 100%;
        }
```

```
        .module li{
            letter-spacing: normal;
            float: left;
            width:480px;
            height: 100%;
            display: inline-block;
            background-repeat: no-repeat;
            margin-left: 3px;
            border-radius: 5%;
            padding-top: 17%;
            padding-bottom: 20%;
            text-align: center;
        }
        .module li:hover{

            transition: 0.5s;
        }
        #author{
            background-color: #D2F1D0;
            background-position: 50% 50%;
            background-size: 300% 100%;
            color: #6C9B69;
        }
        #paper{
            background-color: #FFFFCF;
            background-position: 50% 50%;
            background-size: 300% 100%;
            color: #BABA71;
        }
        #conference{
            background-color: #CFF4FF;
            background-position: 100% 50%;
            background-size: 300% 100%;
            color: #7999A2;
        }
    </style>
```

Above is a part of the whole **css** file, it starts with the tag "style". And you can edit all the style of your text, if you classify your element before. For example, you can change your font-color or font-family easily, you can even decide the composing of the whole page. Shortly speaking, **css** is a free area for "artists" to achieve their all imagination toward the web design.

Let's suppose a class named *paper*, then you can edit the members of the *paper* by **.paper h1** for instance. **.** means it's a name of a class, and on the contrary, if you

use **#**, then it means it's an id.

You must mention **:hover**, it'a a simple way to make some hover effect, especially for hyperlink.

And for hyperlink, it has four attributes that are link, visited, hover, active. And their meanings are just as their names.

So in the conclude, **css** is a tool to help you construct the main structure of your website.

1.3.2 JavaScript

JavaScript is in charge of the information transmission and all the dynamic effect. So in the process of my coding, it's the most important part.

```
<script>
    var page=1;
    $(document).ready(function(){

        var url=location.search;
        url=url.substr(1);
        key_word=url.split('=')[1];
        showPaper(key_word,0);
        //$(".title").css("opacity",0);
        $(".title").animate({opacity:1},800);
        //$(".title").animate({opacity:'1'},1000);
        $(".title").find("h1").fadeIn(1000);

        $("#next").click(function(){
            page++;
            showPaper(key_word,page);
        });
        $("#preview").click(function(){
            page--;
            showPaper(key_word,page)
        });
        $(".page_turn").click(function(){
            if ($(this).find("h3").text()=="<" || $(this).find("h3
                ").text()==">") return;
            page=parseInt($(this).find("h3").text());
            showPaper(key_word,page);
        });
        $("div.guide").mouseover(function(){
            $("div.guide").find("a").css("color","black");
        });
```

```
        $("div.guide").mouseout(function(){
            $("div.guide").find("a").css("color","white");
        });
        $("a.guide_turn").mouseover(function(){
            $(this).animate({fontSize:"+=3px"},"fast");
        });
        $("a.guide_turn").mouseout(function(){
            $(this).animate({fontSize:"-=3px"},"fast");
        });
    });
</script>
```

Here, **jQuery** is the most powerful tool to simplify your work. It's a package of **JavaScript**, which means it contains most useful functions to help you much. For example, the selector of **jQuery** can help you find your target element fast, if you mark them before. The format is similar to the **css**.

```
$(".page_turn").click(function(){
        if ($(this).find("h3").text()=="<" || $(this).find("h3").
           text()==">") return;
        page=parseInt($(this).find("h3").text());
        showPaper(key_word,page);
    });
```

At first, you can find your target element just by its id or class name, and then, according to the DOM tree you construct, you can find its parents and children easily, just as $(this).child() or .parent(). After that, you can change the content of the element with ways including **text(), hide(), show(), css(), attr()** and so on to change all any attributes you want to change.

Then, if you need the data that are stored in the database, you can call the corresponding PHP file to get them.

```
function Recommand(key_word)
{
    var xmlhttp;
    if (window.XMLHttpRequest)
    {
        xmlhttp=new XMLHttpRequest();
    }else
    {
        xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
    xmlhttp.onreadystatechange=function()
```

```
    {
        if (xmlhttp.readyState==4 && xmlhttp.status==200)
        {
            //alert(xmlhttp.responseText);
            var data=eval(xmlhttp.responseText);
            for (i=1;i<=4 && i<=data.length;i++)
            {
                var str="li#li_"+i;
                var title=data[i-1]["Title"];
                var num=parseInt(20*Math.random()+1).toString();
                var st="images/paper/paper_"+num+".jpg";
                $("div.recommand").find(str).hide();
                $("div.recommand").find(str).find("img#img").show
                    ();
                $("div.recommand").find(str).find("img#img").attr(
                    "src",st);
                $("div.recommand").find(str).find("h").text(title)
                    ;
                $("div.recommand").find(str).show();

                St='paper.html?ID='+data[i-1]["PaperID"]+"&img="+
                    num;
                $("div.recommand").find(str).find("a").attr("href"
                    ,St);
                $("div.recommand").find(str).find("a").attr("title
                    ",title);
            }
            for (i=data.length+1;i<=4;i++)
            {
                var str="li#li_"+i;
                $("div.recommand").find(str).hide();
                $("#next").hide();
            }
        }
    }
    xmlhttp.open("GET","paper/recommand.php?ID="+key_word,true);
    xmlhttp.send();
    return;
}
```

With the help of XMLhttpRequest, we can send request to the php file and then, by **GET**, we just can get the data we want and use them to make some change to the original page.

1.4 Presentation

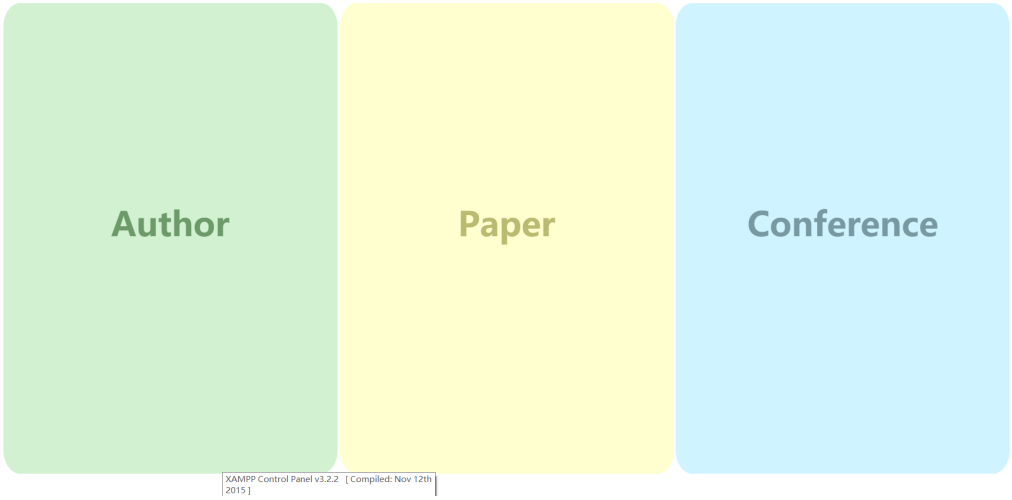

Figure 1.1: home

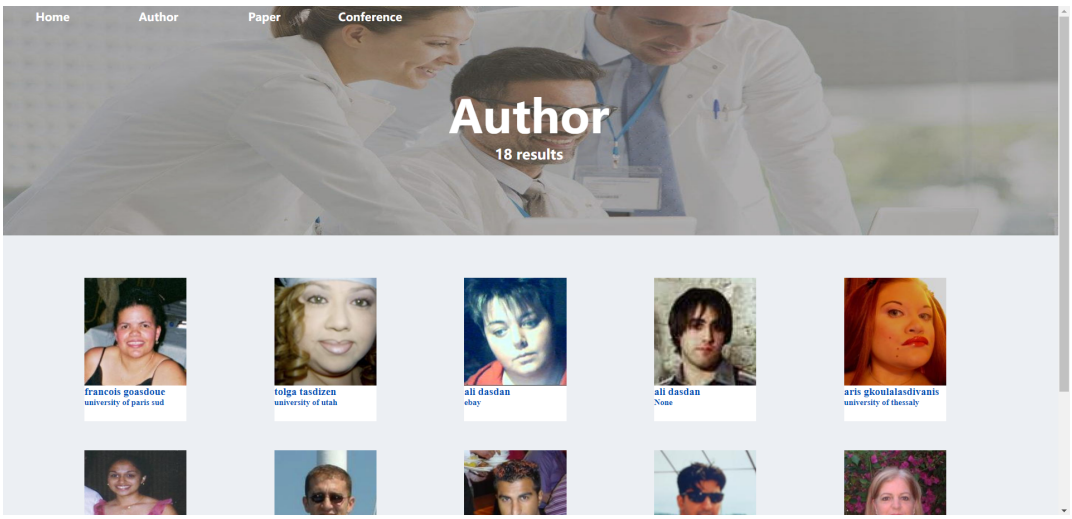

Figure 1.2: result

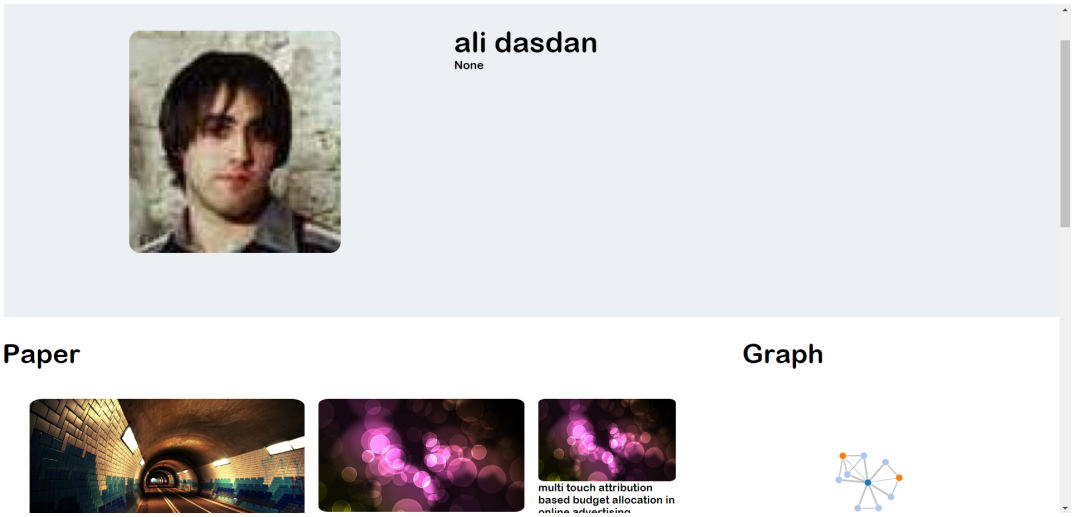

Figure 1.3: author

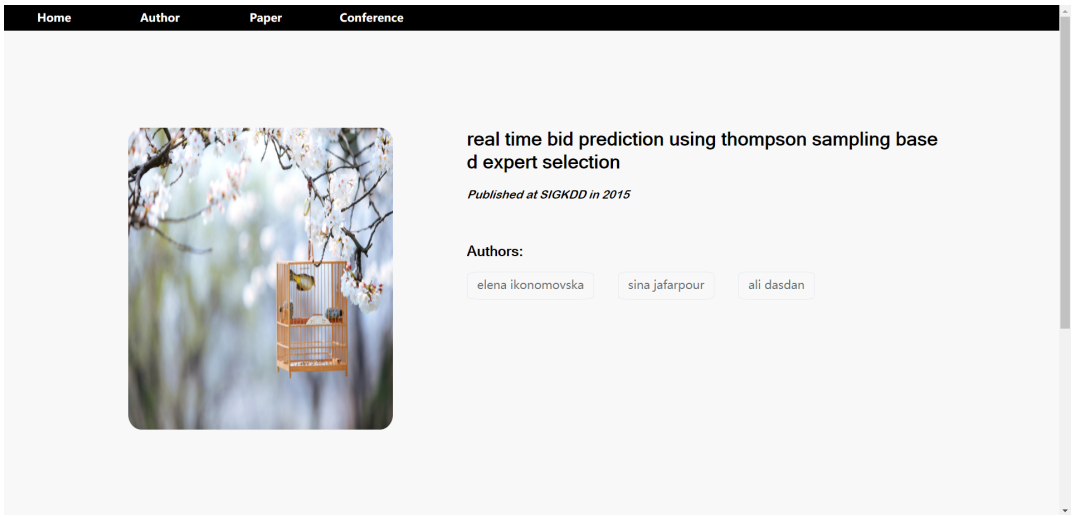

Figure 1.4: paper

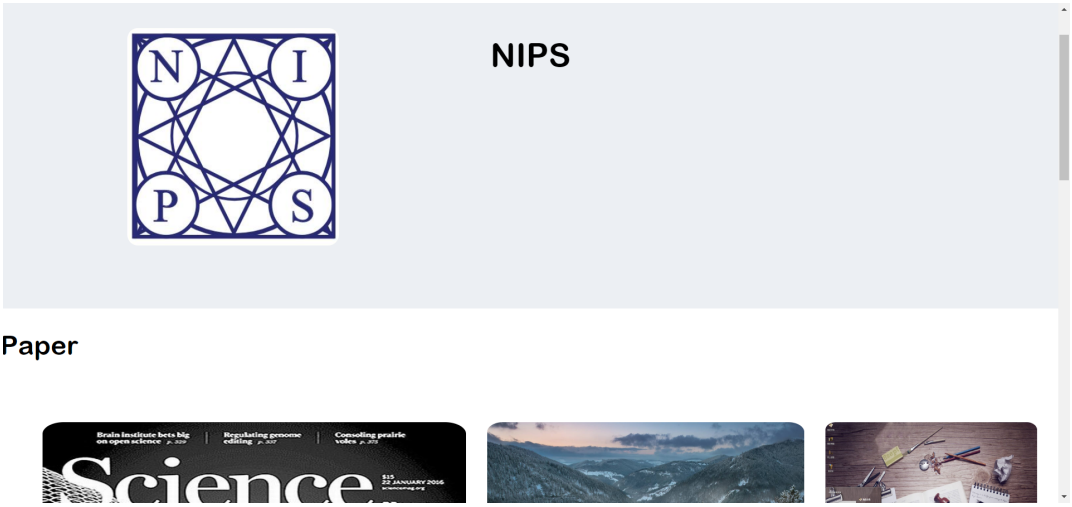

Figure 1.5: conference

Chapter 2

All about paper and Some visualization

Here I made data transmission on paper pages. And I pecdict articles that reader may be interested.And I made visualization in author page.

2.1 Introduction

Search results pages are added to paper title .Adding paper pages to the web site.I build transmission gates of paper and author And I also do something about predict articles that the reader may be interested.Finally I made a preliminary visualization of author page

2.2 process

2.2.1 search pages

Here I just add a paper search similar to author search, and exist on the other page independently of author search. When you need to search for paper, point it out

This is so easy ,It's just a small change in the search page of author.

The code we use is put as below.

```
<form action="paper4.php" method="get">
<p align='center' style="font-size: 30px">
    Papertitle:<input type="text" id="Title" name="Title" style="
       height:20px;width:1000px;font-size: 20px">
```

```
    <button type="submit" class="ui-button ui-corner-all ui-weight
       " style="background: #3366ff;color:white;border:none;height
       :30px;">Search</button>
</p>
</form>
<script src="jquery.js"></script>
<script src="jquery-ui.js"></script>
 <script>
        $(function(){
            $("#Title").autocomplete({
                source: "hint2.php",

            });
        });
```

After that is the modification of the hint page. The hint page should process and complete the home page after transmitting the relevant information of title. This involves some database statements and linkage with the JS of the home page.

And the statement of the database is more important .

Example 2.1 (Data Sample)

```
 {
  $query = mysql_query('SELECT papers.Title
    FROM papers
    WHERE papers.Title like '%".$content.'%" ');
  }
```

2.3 paper page

On the paper page, I first get the paper title that is transferred from the hint page, and the other information I need through this title to select, such as pubulishyear, paperid, author .

First I connect with datebase

```
php $ptitle = $_GET["Title"];
$dbhost = 'localhost:3306';
$dbuser = 'root';
$dbpass = '';
```

```
$conn = mysqli_connect($dbhost, $dbuser, $dbpass);

mysqli_query($conn , "set names utf8");
```

Then I Then I simply searched the papertitle through get and searched for things like PaperID, PaperPublishYear, ConferenceName and so on, and put them in the form to make my page clearer.

And the variable '$params' is much like the SQL statements.

```
$retval_1=mysqli_query($conn,$sql_1);
echo '<table border="1" id="newpage"><tr><td>title</td><td>ID</td
   ><td>YEAR</td><td>CONFERENCE</td><td>AUTHORNAME</td></tr>';

while ($row_1=mysqli_fetch_array($retval_1, MYSQL_ASSOC))
{
    $ID=$row_1['ConferenceID'];
    $sql_2="select ConferenceName from Conferences where
       ConferenceID='$ID'";
    $retval_2=mysqli_query($conn, $sql_2);
    $row_2=mysqli_fetch_array($retval_2, MYSQL_ASSOC);
    echo "<tr><td>{$row_1['Title']}</a></td>".
         "<td>{$row_1['PaperID']} </td> ".
         "<td>{$row_1['PaperPublishYear']} </td> ".
         "<td>{$row_2['ConferenceName']} </td> ".
         "<td>";
```

Then I get more information from the paperID I get, such as looking for authored in paper_author_affiliation, and looking for authorName in authors through authorid. Finally, put them in the list and create hyperlinks from author name to author pages.

```
$ID2=$row_1['PaperID'];
    $sql3="select AuthorID from paper_author_affiliation
           where PaperID='$ID2'
           order by Authorsequence";
    $retval3 = mysqli_query( $conn, $sql3 );
    if(! $retval3 )
    {
        die('Failed to input the data: ' . mysqli_error($conn));
    }
    $num=1;
    while($row3 = mysqli_fetch_array($retval3, MYSQL_ASSOC))
    {
        echo $num;
```

```
        $num=$num+1;
        echo ": ";
        $AID=$row3['AuthorID'];
        $sql5="select AuthorName from authors
               where AuthorID='$AID'";
        $retval5 = mysqli_query( $conn, $sql5 );
        if(! $retval5 )
        {
            die('Failed to input the data: ' . mysqli_error($conn)
               );
        }
        $row5 = mysqli_fetch_array($retval5, MYSQL_ASSOC);
        $new_auname=$row5['AuthorName'];
        echo "<a href='author.php?ID=".$AID."'>".$new_auname."</a>
           ";
        echo ".  ";
    }
```

2.3.1 pecdict articles

In the process of recommending, I adopted a paper that recommends the paper and the way of the paper quoted by the paper. The paper quoted in this paper may have a more detailed deduction and proof of some of the basic points in the paper, and the paper quoted by this paper may have further expansion of the articles described in the paper. Exhibition. I then sort these papers and select the two highest "heat", where I use the frequency of a paper cited, that is, the number of times used to show its heat. The number of times a paper can be quoted can reflect its degree of being used by others, so sorting through this method is reasonable.

```
$sql6="select papers.Title,papers.PaperID
    from paper_reference,papers
    where papers.PaperID=paper_reference.PaperID and
       paper_reference.ReferenceID='$ID2' limit 2
    order by Referred";
    $retval6 = mysqli_query( $conn, $sql6 );
    if(! $retval6 )
    {
        die('none: ' . mysqli_error($conn));
    }
    $row_6=mysqli_fetch_array($retval_6, MYSQL_ASSOC);
    echo "</td>";
    $sql7="select papers.Title,papers.PaperID
    from paper_reference,papers
    where papers.PaperID=paper_reference.ReferenceID and
       paper_reference.PaperID='$ID' limit 2
    order by Referred";
```

```
    $retval7 = mysqli_query( $conn, $sql7 );
    if(! $retval7 )
    {
        die('none: ' . mysqli_error($conn));
    }
    $row_7=mysqli_fetch_array($retval_7, MYSQL_ASSOC);
```

The other thing I did was to optimize the database during this process, because I frequently invoked the number of references to express the heat, so the number of references was invoked as a separate column, which made my program faster and more efficient, so I updated the database and added a list of "refered". This indicates the heat, so that I can improve the convenience and efficiency of my program when I call.

```
if (!$conn) {
    die("Failed to connect: " . mysqli_connect_error());
}

//$sql="select Referred,PaperID from paper_reference;";
//$result = mysqli_query($conn, $sql);
$paper=mysqli_query($conn,'select Referred,PaperID from
   paper_reference');
while ($temp=mysqli_fetch_array($paper)){
        if ($temp['Referred']==NULL)
        {
            $count=mysqli_query($conn,'SELECT * From
               paper_reference WHERE ReferenceID="'.$temp['PaperID
               '].'"');
            $num=mysqli_num_rows($count);
            if ($num==0) $st="0"; else $st=(string)$num;

            mysqli_query($conn,'UPDATE paper SET Referred='.$st.'
               WHERE PaperID="'.$temp['PaperID'].'"');
        }else
        {
            $num=$temp['Referred'];
        }
}
```

2.3.2 visualization

In each author page, add a visual map of this scholar and all his collaborators. That is to find all the partners of author, and find all the cooperative relationships among these scholars, then visualize the network structure between these points.

You can use the Force-Directed Graph framework in reference link 1. At the same time, the point of the current scholar and his collaborator should be distinguished from different colors.

In the process of visualization, I created the author as a one point and searched through mysql, search out all his papers, and find out all the partners who have a partnership with him through the paper, and connect the author to his partner.

The first is to call the appropriate JS file

```
</style>
<svg width="960" height="600"></svg>
<script src="https://d3js.org/d3.v4.min.js"></script>
<script>
```

Then create node and line connections

```
\var svg = d3.select("svg"),
    width = +svg.attr("width"),
    height = +svg.attr("height");

var color = d3.scaleOrdinal(d3.schemeCategory20);

var simulation = d3.forceSimulation()
    .force("link", d3.forceLink().id(function(d) { return d.id; })
        )
    .force("charge", d3.forceManyBody())
    .force("center", d3.forceCenter(width / 2, height / 2));

d3.json("achieve.json", function(error, graph) {
  if (error) throw error;

  var link = svg.append("g")
      .attr("class", "links")
    .selectAll("line")
    .data(graph.links)
    .enter().append("line")
      .attr("stroke-width", function(d) { return Math.sqrt(d.value
          ); });

  var node = svg.append("g")
      .attr("class", "nodes")
    .selectAll("circle")
    .data(graph.nodes)
    .enter().append("circle")
```

```
      .attr("r", 5)
      .attr("fill", function(d) { return color(d.group); })
      .call(d3.drag()
          .on("start", dragstarted)
          .on("drag", dragged)
          .on("end", dragended));

  node.append("title")
      .text(function(d) { return d.id; });

  simulation
      .nodes(graph.nodes)
      .on("tick", ticked);

  simulation.force("link")
      .links(graph.links);

  function ticked() {
    link
        .attr("x1", function(d) { return d.source.x; })
        .attr("y1", function(d) { return d.source.y; })
        .attr("x2", function(d) { return d.target.x; })
        .attr("y2", function(d) { return d.target.y; });

    node
        .attr("cx", function(d) { return d.x; })
        .attr("cy", function(d) { return d.y; });
  }
});

function dragstarted(d) {
  if (!d3.event.active) simulation.alphaTarget(0.3).restart();
  d.fx = d.x;
  d.fy = d.y;
}

function dragged(d) {
  d.fx = d3.event.x;
  d.fy = d3.event.y;
}

function dragended(d) {
  if (!d3.event.active) simulation.alphaTarget(0);
  d.fx = null;
  d.fy = null;
}
```

Then connect to the database and search for relevant authors through mysql.

```
mysqli_query($conn , "set names utf8");

$sql_0="select AUTHORID from authors where AUTHORID in
        (select AuthorID from paper_author_affiliation
         where PaperID in(select PaperID from
             paper_author_affiliation
         where AuthorID='$AUTHORID'))";
mysqli_select_db($conn, 'mymap');
```

After the ID is equal to authored, it is divided into group1 and group2, and then it is processed separately. Finally, the points are presented on behalf of each author and connected with the line to form a complete visualization map.

```
foreach($arr_id as $ids){
    array_shift($arr_id);
    foreach($arr_id as $ids_2){
        $dbhost = 'localhost';
        $dbuser = 'root';
        $dbpass = '';
        $conn = mysqli_connect($dbhost, $dbuser, $dbpass);
        if(! $conn )
        {
            die('Failed to connect: ' . mysqli_error($conn));
        }
        mysqli_query($conn , "set names utf8");

        $sql_11="select PAPERID from paper_author_affiliation
                where AuthorID='$ids'
                and PAPERID in
                (select PAPERID from paper_author_affiliation
                   where AuthorID='$ids_2')";
        mysqli_select_db($conn, 'mymap');
        $retval_11 = mysqli_query( $conn, $sql_11);
        if(! $retval_11 )
        {
            die('Failed to input the data: ' . mysqli_error($conn)
               );
        }
        $row_11=mysqli_fetch_array($retval_11, MYSQL_ASSOC);
        if($row_11)
        {
            $sql_12="select AUTHORNAME from authors where AUTHORID
               ='$ids'";
            mysqli_select_db($conn, 'mymap');
            $retval_12 = mysqli_query( $conn, $sql_12);
```

```
            $row_12=mysqli_fetch_array($retval_12, MYSQL_ASSOC);
            $NAME1=$row_12['AUTHORNAME'];

            $sql_13="select AUTHORNAME from authors where AUTHORID
                ='$ids_2'";
            mysqli_select_db($conn, 'mymap');
            $retval_13 = mysqli_query( $conn, $sql_13);
            $row_13=mysqli_fetch_array($retval_13, MYSQL_ASSOC);
            $NAME2=$row_13['AUTHORNAME'];
            $arr_tmp1=array("source"=>$NAME1,"target"=>$NAME2,"
                value"=>1);
            array_push($arr_link,$arr_tmp1);
        }
    }
```

The last step is how to do, how to connect and their physical properties.

```
$arr_all=array("nodes"=>$arr_node,"links"=>$arr_link);
$jfile=fopen("test.json","w")or die("unable to open file.");
$json_all=json_encode($arr_all);
fwrite($jfile,$json_all);
fclose($jfile);

<head>
    <link href="jquery-ui.css" rel="stylesheet">
    <style>
    body{
        font-family: "Trebuchet MS", sans-serif;
        margin: 50px;
    }
    a:link {
        color:#000000;
        text-decoration:underline;
        font-weight: bold;
    }
    a:visited {
        color:#000000;
        text-decoration:underline;
        font-weight: bold;
    }
    a:hover {
        color:#000000;
        text-decoration:underline;
        font-weight: bold;
    }
    a:active {
```

```
        color:#000000;
        text-decoration:none;
        font-weight: bold;
    }
    p.thick{
        font-weight: bold;
    }
    .demoHeaders {
        margin-top: 2em;
    }
    #dialog-link {
        padding: .4em 1em .4em 20px;
        text-decoration: none;
        position: relative;
    }
    #dialog-link span.ui-icon {
        margin: 0 5px 0 0;
        position: absolute;
        left: .2em;
        top: 50%;
        margin-top: -8px;
    }
    #icons {
        margin: 0;
        padding: 0;
    }
    #icons li {
        margin: 2px;
        position: relative;
        padding: 4px 0;
        cursor: pointer;
        float: left;
        list-style: none;
    }
    #icons span.ui-icon {
        float: left;
        margin: 0 4px;
    }
    .fakewindowcontain .ui-widget-overlay {
        position: absolute;
    }
    select {
        width: 200px;
    }
```

After that, Ma Jiaqi optimized the visualization and increased the prediction and performance of teacher-student relationship. I don't give a detailed account of it here.

2.4 Conclusion

This is all I do .I am jiangzekun.Thanks for the help of my partner.And Thank the teacher for the review .

Chapter 3

Recommendation

3.1 Introduction

In our daily life,if we want to search something we need, we use search engines like Baidu or Google.After we input the contents,we can get exactly what we want.At the same time,the result pages usually contain recommendation part on the margin of the page or at the bottom of the page.These recommendation contents have certain relationship with what we want to search.Maybe the users are interested in the contents in recommendation part.And recommendation part can help users to find other similar information.

3.2 How to recommend other papers

In our project,we add recommendation part in our author.php page.In our original author page,it contains certain author's papers,affiliation.If we want to recommend other papers to the users,what we want to recommend should relate to this author,this author's papers and this author's affiliation.So we design five ways to recommend.We cannot only rely on one or two recommendation methods.More recommendation ways can ensure the diversity of contents and avoid some methods getting no contents.Here are the ways we how to recommend:

3.2.1 This author's other papers

In some cases,one author's papers will not all show at the current pages.Maybe the users are most interested in this author.So we directly recommend this author's other papers.For example,if we now view the page 1 of certain author,this page shows 10 papes,and we maybe recommend the papers in page 100 which shows another 10 papers.

3.2.2 Other papers which cite this author's papers

When we enter in author page,maybe we are interested in this author's papers.So we can recommend papers with similar topics and theme to users.How can we make sure of the similarity or correlation.We can utilize the citation.Usually,one paper cites the same or similar topic papers.

3.2.3 This author's co-workers' papers

One paper may have many authors,they work together to finish this paper.And these co-workers may research the same filed as this author researches.So their papers may focus on the same topics

3.2.4 Other papers published at the same conference in the same year

If we don't have something to recommend,we can turn to a larger scope to get some contents.Maybe one paper has certain relationship with the papers which were published in the same year at the same conference

3.3 How to realize it

1.This author's other papers We count the total number of one author's papers,if this number is over 10,we can continue.And we recommend papers which are not at the page we view now.

2.Other papers which cite this author's papers We use reference id to get.We get paper id of one author

3.This author's co-workers' papers In papers file,one paper has many authors.We use their author id to get their paper id and paper title.

4.Other papers published at the same conference in the same year One paper has publish year and the affiliation id.We randomly get a paper in the same year and conference.

Chapter 4

Elasticsearch

Since the original MySQL database is not efficient enough for query, an emergency of chaning the search method is called for. After looking up for a lot of information, we choose ELASTICSEARCH as our search engine.

4.1 Introduction

Elasticsearch is a search engine based on Lucene. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents. Elasticsearch is developed in Java and is released as open source under the terms of the Apache License. Official clients are available in Java, .NET (C#), PHP, Python, Apache Groovy, Ruby and many other languages. According to the [**DB-Engines ranking**], Elasticsearch is the most popular enterprise search engine.

4.2 Preparation

4.2.1 Installation

For installation, besides elasticsearch itself, JAVA is at least required. To be careful, we can't install the latest version, for it will be incompatible with some plug-in components we will use later.

4.2.2 Data Import

Then, to begin with, we should import our database from MySQL to elasticsearch, which requires a JAVA API called JDBC. Its full name is Java DataBase Connectivity, a Java-based data access technology to create and execute SQL statements.

Under the path of elasticsearch-jdbc, we create a shell file, where we write our SQL statements. (mysql-import.sh)

Here we take import PAPER data as an example, and the code we use is put as below.

```
#!/bin/sh
bin=$JDBC_IMPORTER_HOME/bin
lib=$JDBC_IMPORTER_HOME/lib //the path of JDBC
echo '{
"type" : "jdbc",
"jdbc": {
"elasticsearch.autodiscover":true,
"elasticsearch.cluster":"elasticsearch",
"url":"jdbc:mysql://127.0.0.1:3306/test",
"user":"root",
"password":"",
"sql":"select *, PaperID as _id from papers",
"elasticsearch" : {
  "host" : "localhost",
  "port" : 9300
},
"index" : "main_db",
"type" : "papers"
}
}'| java \
  -cp "${lib}/*" \
  -Dlog4j.configurationFile=${bin}/log4j2.xml \
  org.xbib.tools.Runner \
  org.xbib.tools.JDBCImporter
```

Then we use Kit to execute the file and the data in the table is imported to elasticsearch in JSON form. **??**.

Example 4.1 (Data Sample)

```
{
 "took" : 77,
 "timed_out" : false,
 "_shards" : {
   "total" : 5,
   "successful" : 5,
   "failed" : 0
 },
 "hits" : {
```

```
  "total" : 98215,
  "max_score" : 1.0,
  "hits" : [ {
    "_index" : "main_db",
    "_type" : "papers",
    "_id" : "76834DAC",
    "_score" : 1.0,
    "_source" : {
      "PaperID" : "76834DAC",
      "Title" : "methods study for the relocation of visual
                 information in central scotoma cases",
      "PaperPublishYear" : 2005,
      "ConferenceID" : "43319DD4"
    }
  }, {
    "_index" : "main_db",
    "_type" : "papers",
    "_id" : "7FB3ECEA",
    "_score" : 1.0,
    "_source" : {
      "PaperID" : "7FB3ECEA",
      "Title" : "extracting viewpoint invariance relations
                 using fuzzy sets",
      "PaperPublishYear" : 1992,
      "ConferenceID" : "43319DD4"
    }
  }, {
    "_index" : "main_db",
    "_type" : "papers",
    "_id" : "0119F6B8",
    "_score" : 1.0,
    "_source" : {
      "PaperID" : "0119F6B8",
      "Title" : "self optimizing image segmentation system
                 using a genetic algorithm",
      "PaperPublishYear" : 1991,
      "ConferenceID" : "43319DD4"
    }
  }, {
    "_index" : "main_db",
    "_type" : "papers",
    "_id" : "810588E8",
    "_score" : 1.0,
    "_source" : {
      "PaperID" : "810588E8",
      "Title" : "the dynamic beamformer",
```

```
        "PaperPublishYear" : 2012,
        "ConferenceID" : "43319DD4"
      }
    }, {
      "_index" : "main_db",
      "_type" : "papers",
      "_id" : "79FADF4D",
      "_score" : 1.0,
      "_source" : {
        "PaperID" : "79FADF4D",
        "Title" : "frequency and spatial pooling of visual
                   differences for still image quality assessment
                      ",
        "PaperPublishYear" : 2000,
        "ConferenceID" : "43319DD4"
      }
    } ]
  }
}
```

4.3 Complementation Procedure

After that, we come to the complementation procedure - the search function. We first need to connect PHP to elasticsearch. And this time we need Composer. It is an application-level package manager for the PHP programing language that provides a standard format for managing dependencies of PHP software and required libraries.

We are required to build a new folder and install Composer in it in order to generate an elasticsearch-PHP framework. Later on, we create a file 'composer.json', whose content is:

```
{
"require":{
    "elasticsearch/elasticsearch" : "~2.0''
    }
}
```

And under the folder path, we execute the order:

```
php composer.phar install --no-dev
```

Now, we have already completed the PHP framework. Then we can substitute the query statements in the original codes. In the report, I will only take the paper website as example. The demand is: we fuzzy search the paper information according to the input text, and then sort the results according to each paper's cited times in descending order. Besides, we show each paper's author name and the names are displayed according to the author sequence.(es-test.php)

First we import the autoload framework.

```
require_once('\test\vendor\autoload.php');
```

After that, we create a new client of elasticsearch.

```
$client = new Elasticsearch\Client($params);
```

And the variable '$params' is much like the SQL statements.

```
$params = array(
    'index' => 'main_db', //index functions as database
    'type' => 'papers', //type functions as each table
    'body' => array(
        'size' => 10000,    //the default size is 10
        'query' => array(
            'match' => array(
                'Title' => $x."?*" //'$x' symbolizes the input
                    text and '?*' helps achieve the fuzzy search
            )
        ),
    )
);
```

Then, we use search format to gain the result.

```
$rtn0 = $client->search($params);
```

Print our result in a formal form, and we can get these multi-dimension arrays:

Because we need sort it, however, elasticsearch has difficulty in query data from two or more types. The solution is that we will restore each PaperID, according to which we do the second query and get the cited times for each paper.

```
$num = $rtn0["hits"]["total"]; //the query size
$result = array();  //to restore the result
for ($i=0;$i<$num;$i++){
    $params = array(
    'index' => 'main_db',
    'type' => 'paper_reference',
    'body' => array(
        'size' => 1000,
        'query' => array(
            'match' => array(
                'ReferenceID' => $rtn0["hits"]["hits"][$i]["
                    _source"]["PaperID"]
                )
            )
        )
    );
    $rtn = $client->search($params);
    array_push($result,array($rtn["hits"]["total"],$rtn0["hits"]["
        hits"][$i]["_source"]["Title"],$rtn0["hits"]["hits"][$i]["
        _source"]["PaperID"])); //restore each paper's title, cited
         times and PaperID
}
```

Then we need to sort it.

```
array_multisort($result,SORT_DESC);
```

it's time to query the author of each paper. Here we only display the top 10 results.

```
$i = 0;
echo '<div id="text">';
while($i<10){
    echo $result[$i][0]."\t".$result[$i][1];
    echo "<br>";
    $params = array(
        'index' => 'main_db',
        'type' => 'paper_author_affiliation',
        'body' => array(
            'query' => array(
                'match' => array(
                    'PaperID' => $result[$i][2]
                )
            ),
            'sort' => array(
```

```
                'AuthorSequence' => array(
                    'order' => 'asc' //key is we display the
                        result in AuthorSequence's ascending order
                )
            )
        )
    );
    $rtn2 = $client->search($params);
    $num2 = $rtn2["hits"]["total"];
    for ($j=0; $j<$num2; $j++){
        print_r($rtn2["hits"]["hits"][$j]["_source"]["
            AuthorSequence"]);
        echo "\t";
        $params = array(
            'index' => 'main_db',
            'type' => 'authors',
            'body' => array(
                'query' => array(
                    'match' => array(
                        'AuthorID' => $rtn2["hits"]["hits"][$j]["
                            _source"]["AuthorID"]
                    )
                )
            )
        );
    $rtn3 = $client->search($params);
    print_r($rtn3["hits"]["hits"][0]["_source"]["AuthorName"]);
    echo "<br>";
    }
    echo "<br>";
    $i++;
}
```

4.4 Conclusion

Just query "technique" as an example, and record the search time:

To make a comparison, if we use the original database in MySQL, it takes more than 60 seconds to accomplish the 'inner join' order.

So, obviously, we can draw a safe conclusion that elasticsearch has an advantage in search speed over MySQL.

```
Array
(
    [took] => 1
    [timed_out] =>
    [_shards] => Array
        (
            [total] => 5
            [successful] => 5
            [failed] => 0
        )

    [hits] => Array
        (
            [total] => 251
            [max_score] => 4.4674497
            [hits] => Array
                (
                    [0] => Array
                        (
                            [_index] => main_db
                            [_type] => papers
                            [_id] => 8096887C
                            [_score] => 4.4674497
                            [_source] => Array
                                (
                                    [PaperID] => 8096887C
                                    [Title] => the cordic computing technique
                                    [PaperPublishYear] => 1959
                                    [ConferenceID] => 46DAB993
                                )

                        )

                    [1] => Array
                        (
                            [_index] => main_db
                            [_type] => papers
                            [_id] => 754F6985
                            [_score] => 4.35102
                            [_source] => Array
                                (
                                    [PaperID] => 754F6985
                                    [Title] => the electrographic recording technique
                                    [PaperPublishYear] => 1955
                                    [ConferenceID] => 46DAB993
                                )

                        )

                    [2] => Array
                        (
                            [_index] => main_db
                            [_type] => papers
                            [_id] => 80CCF44B
                            [_score] => 3.3505874
                            [_source] => Array
                                (
                                    [PaperID] => 80CCF44B
                                    [Title] => a robust hybrid iris localization technique
                                    [PaperPublishYear] => 2009
                                    [ConferenceID] => 436976F3
                                )

                        )
```

Figure 4.1: best

90 an iterative image registration technique with an application to stereo vision
1 bruce d lucas
2 takeo kanade

15 an efficient and accurate camera calibration technique for 3d machine vision
1 roger y tsai

6 a spectral technique for correspondence problems using pairwise constraints
1 marius leordeanu
2 martial hebert

5 sentence disambiguation by a shift reduce parsing technique
1 stuart m shieber

5 model based curve evolution technique for image segmentation
1 andy tsai
2 anthony yezzi
3 william m wells
4 clare m c tempany
5 don m tucker
6 alan fan
7 w e grimson
8 alan s willsky

5 a multi resolution technique for comparing images using the hausdorff distance
1 daniel p huttenlocher
2 william j rucklidge

4 skeletonization a technique for trimming the fat from a network via relevance assessment
1 michael c mozer
2 paul smolensky

4 em dd an improved multiple instance learning technique
1 qi zhang
2 sally a goldman

3 automated scoring using a hybrid feature identification technique
1 jill burstein
2 karen kukich
3 susanne wolff
4 chi lu
5 martin chodorow
6 lisa c bradenharder
7 mary dee harris

3 an inference technique for integrating knowledge from disparate sources
1 thomas d garvey
2 john d lowrance
3 martin a fischler

本次搜索用时3.3919470310211秒

Figure 4.2: best