

# Experiment Report

王一霖\_范廷宇\_闻天麟\_顾诗轩

2018年6月19日

# 目录

<b>1 Program Structure and Basic functions</b>	<b>4</b>
1.1 Program Structure . . . . .	4
1.2 Basic Functions . . . . .	4
1.2.1 home pages . . . . .	4
1.2.2 Autocomplete Function . . . . .	5
1.2.3 Menu of searching results . . . . .	6
1.2.4 Paging Ability . . . . .	8
1.2.5 Final Result Displaying . . . . .	9
<b>2 Recommendation for papers</b>	<b>9</b>
2.1 Problem description . . . . .	9
2.2 Problem analysis . . . . .	10
2.3 Solutions and codes . . . . .	10
<b>3 Teachers and Students Relationship Trees</b>	<b>15</b>
3.1 Backups . . . . .	15
3.2 Codes . . . . .	15
3.2.1 Problems of origin codes . . . . .	21
3.3 Improvements to the codes . . . . .	21
3.3.1 Python Code . . . . .	22
3.3.2 PHP code . . . . .	30
3.3.3 Java Script code . . . . .	31
3.3.4 ECharts code . . . . .	32
3.4 Possible Improvement . . . . .	35
3.4.1 Running Time . . . . .	35
3.4.2 Deeper Trees . . . . .	35
<b>4 Beautification of the Webpage</b>	<b>35</b>
4.1 PROCEDURE . . . . .	37
4.1.1 The dynamic background designing. . . . .	37
4.1.2 The navigation designing. . . . .	42
4.1.3 Modifications of typeface . . . . .	43
4.2 THE RESULT DISPLAY . . . . .	48

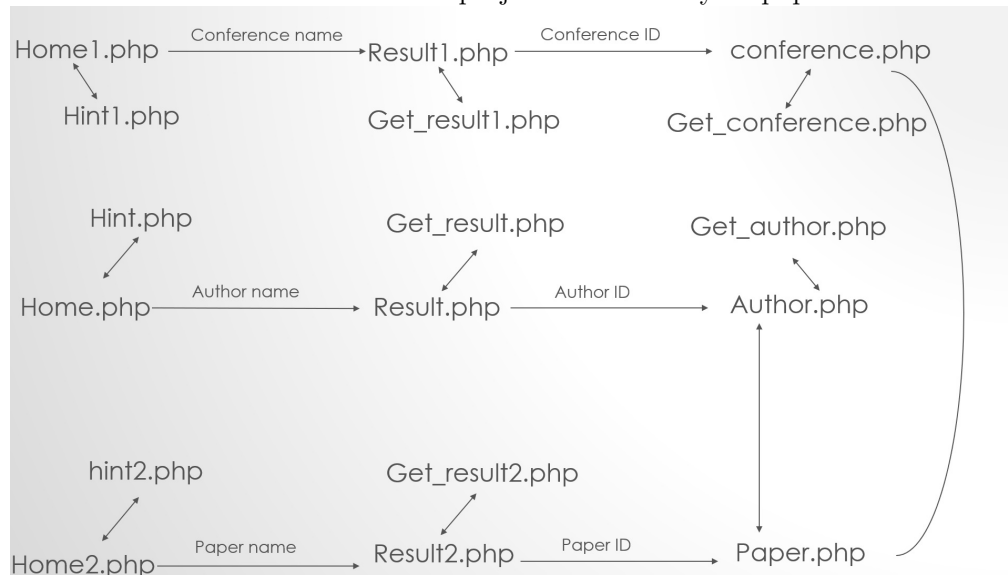


# 1 Program Structure and Basic functions

Done by Fan Tingyu.

## 1.1 Program Structure

The basic functions of our final project is realized by 12 php files:



## 1.2 Basic Functions

### 1.2.1 home pages

We designed 3 home pages respectively corresponding to author-search, paper-search, conference-search. Each home page has an input box with autocomplete function.

Use the code of author-search(home.php) for example:

Form(containing the input box and submit button):

```
1 <form action=result.php method="get">
2 <input style="font-size:20px" type="text" id="key" name="name"/>
3 <input style="font-size:20px" type="submit" name="submit"
4 value="Submit">
```

```
5 </form>
```

Autocomplete function(source file hint.php):

```
1 <script type="text/javascript">
2 $(function(){
3 $("#key").autocomplete({
4 source: "hint.php",
5 minLength: 2,
6 autoFocus: true
7 });
8 });
9 </script>
```

### 1.2.2 Autocomplete Function

We use 3 php files named hint.php, hint1.php, hint2.php to realize the autocomplete function, which respectively correspond to the autocomplete function of author-search, paper-search, conference-search. Use the code of hint.php for example:

```
1 <?php
2 $conn =new mysqli("localhost", "root", "");
3 mysqli_query($conn,"use homework1");
4 $q = strtolower($_GET["term"]);
5 $query=mysqli_query($conn,"select AuthorName from authors
6 where AuthorName like '$q%' limit 0,10");
7 $result=array();
8 $i=0;
9 While($a=mysqli_fetch_array($query))
10 {
11 $result []= array(
12 'id'=>$i ,
13 'label'=>$a[0] ,
14 );
```

```

15     $i+=1;
16     }
17     echo json_encode( $result );
18     ?>

```

### 1.2.3 Menu of searching results

We use 3 php files named result.php, result1.php, result2.php to design menus which display all the possible results.

Each menu displays 10 objects at one time, with 2 paging buttons to display the former 10 or the next 10 objects.

Use result.php for example:

Body part:

```

1     <p id="table"></p>
2     <button id="formerpage">former page</button>
3     <aaa id="page">page1</aaa>
4     <button id="nextpage">next page</button>

```

We use a function named add() to display the appointed 10 objects on the interface:

```

1     <script >
2     function add(start){
3     $.getJSON(" get_result .php" , {" authorname ":" <?php echo $name;?>" ,
4     " start " : start } , function ( data ) {
5     if ( data . length < 10 ) { max_page = page ; }
6     var text = "<table border = '1' >
7     <th > author name </th > <th > affiliation </th > " ;
8     for ( var i = 0 ; i < data . length ; i ++ ) {
9     text += "<tr > <td id = 'an' + i . toString () + "' > </td >
10    <td id = 'afn' + i . toString () + "' > </td > </tr > " ;
11    }
12    text += "</table > " ;
13    document . getElementById ( " table " ) . innerHTML = text ;

```

```

14   for (var i=0;i<data.length;i++){
15   document.getElementById("an"+i.toString()).innerHTML=
16   "<a href=author.php?AuthorID="+data[i].ai+">" +data[i].an+"</a>";
17   document.getElementById("afn"+i.toString()).innerHTML=data[i].afn;
18   }
19   });
20   }
21   </script >

```

Realization of Paging Ability:

```

1   <script >
2   $(document).ready(function(){
3   $("#nextpage").click(function(){
4   if (page<max_page){
5   page+=1;
6   document.getElementById("page").innerHTML=
7   'page'+(page+1).toString();
8   add(page*10);
9   }
10  });
11  });
12  </script >
13  <script >
14  $(document).ready(function(){
15  $("#formerpage").click(function(){
16  if (page>0){
17  page-=1;
18  document.getElementById("page").innerHTML=
19  'page'+(page+1).toString();
20  add(page*10);
21  }
22  });
23  });

```

### 1.2.4 Paging Ability

Our paging ability is realized by 3 php files named get\_result.php, get\_result1.php, get\_result2.php.

Use get\_result.php for example:

```
1  <?php
2  $json_result=array ();
3  $name=$_GET[" papername "];
4  $start=$_GET[" start "];
5  $conn =new mysqli(" localhost", " root", "");
6  mysqli_query($conn," use homework1");
7  $result=mysqli_query($conn ,
8  "select  PaperID , Title , ConferenceID , PaperPublishYear
9  from papers where Title like '%$name%' limit $start ,10");
10
11  while($a=mysqli_fetch_array($result))
12  {
13  $json_result []= array(" pi"=>$a[0] ,
14  " pn"=>$a[1] ,
15  " ci"=>$a[2] ,
16  " cn"=>mysqli_fetch_array(mysqli_query($conn ,
17  "select  ConferenceName from conferences
18  where ConferenceID='$a[2] '"))[0] ,
19  " ppy"=>$a[3] ,
20  );
21  }
22  echo json_encode($json_result);
23  ?>
```



### 1.2.5 Final Result Displaying

After clicking on the links in result.php/result1.php/result2.php, the interface will jump to a page containing all the information about this object.

author.php will show all papers written by this author, and a tree showing the author's teacher-student relationship. get\_author.php is related to its paging ability.

conference.php will show all papers published in this conference. get\_conference.php is related to its paging ability.

paper.php will show the paper published year, authors and the recommended related papers.

The code for author.php, paper.php, conference.php, get\_author.php, get\_conference.php is shown in the source file.

## 2 Recommendation for papers

Done by Wang Yilin - 517030910327.

After almost one month of groping, we finally finished our final project, in order to pursuit for better, we made some change on our version one, both on its appearance and its inside arithmetic.

### 2.1 Problem description

In this problem, I should give some papers for recommendation according to the papera that users searched. However, if we only take the common authors and references as gist, maybe it is not that efficient and clear for users, because the user don't know why these papers are recommended to them. What I want to build is like the result in Baidu: the results recommended usually have the same words as the key words that users have inputted in the search box.

## 2.2 Problem analysis

In this question, we should consider two questions, first, how can I make it possible to find those papers. Second, how can I condense the running time into an acceptable period: After preliminary test, I find that it can be very slow if we decided to recommend papers according to their name, because if we need to compare almost one million papers, which means we need to get connected to our database in MySQL for almost one million times, which may cost a lot of time.

## 2.3 Solutions and codes

Now I am showing you how did I solve this problem.

First, because we need to get a paper id from php, which was searched by the words that users inputed in the search box. And this paper id is all we need in this problem.

In order to make it more concision and more effective, I decided to use python to do the following jobs, which means I should send this id into a python.

How to send paper id and get the result from python

```
1   exec('python 666.py ' . $paper_id , $output);
2   $data=json_decode($output[0],true);
```

In the above codes, we have the paper id named `$paper_id` in a php named paper, by this function, we can send it into a python named 666, then we can receive the results we get and store it in `$output`.

In the follow part, I am showing you how to do this job in python.

```
3  import sys
4  import json
5  import pymysql
6  import datetime
7
8  paperid = sys.argv[1]
```

In this part, we have stored the paper id from php into paperid.

```
14 db = pymysql.connect("localhost", "root", "", "homework1", charset='utf8' )
15 cursor = db.cursor()
```

After trying, we have find that the running time is mostly cost by connecting to MySQL and searching in our database, in order to condense the running time, we decided to build two txt file named paper and paper\_author, we know that it is so fast to open a txt file for python, in this way we can avoid one million times of connecting into once.

```
17 pa=open("paper_author.txt","r",encoding='utf-8')
18 paper2author={}
19 for i in range(0,98215):
20     temlis=pa.readline().split()
21     paper2author[temlis[0]]=temlis[1:]
22 pa.close()
23
24 pt=open("paper.txt","r",encoding='utf-8')
25 p2t={}
26 allpaperid=[]
27 for i in range(0,98215):
28     temlis=pt.readline().split()
29     p2t[temlis[0]]=temlis[1:]
30     allpaperid.append(temlis[0])
31 pt.close()
```

(We already know that we have 98215 papers.)

In above codes, we have got one paper's authors, and a paper's title.

```
36 words=p2t[paperid]
37 a=0
38
39 while True:
40     if a<len(words):
41         if len(words[a])<6:
42             del words[a]
43         else:
44             a+=1
45     else:
```

```

46         break
47     allauthor=paper2author[paperid]

```

Words is the words in the paper's title that we get from php.Allauthor is the authors in the paper's title that we get from php.

However, because some words may be meaningless like "with" and "for", so we should find a way to avoid those papers recommended according to these words, we choose to delete these words. Finally, we choose to delete the words no longer than 6 letters.

```

60     sql6="""select ReferenceID from paper_reference where PaperID='{ }'""".format(
61         paperid )
62     cursor.execute(sql6)
63     result6= cursor.fetchone()
64     if result6!= None:
65         allaff=list(result6)
66     else:
67         allaff=[]

```

It can be a little bit complex to find references, so I still make use of MySQL to find reference.Different from authors and words, we only need to do this once.

Now we begin to compare papers one by one, we only use a for loop to make this. Then for each paper, we get its authors and words in title, then see if this paper is referenced by the paper we got in php.

Now I am showing you how to compare.

```

70     for oneid in allpaperid:
71
72         #sql3="""select AuthorID from paper_author_affiliation where PaperID='{ }'""".
73             format(oneid[0])
74         #cursor.execute(sql3)
75         #result3= cursor.fetchone()
76         authorforcom=paper2author[oneid]
77         #print(authorforcom)
78         couforauthor=0
79         for i in range(0,len(allauthor)):
80             if allauthor[i] in authorforcom:
81                 couforauthor+=1

```

```

81
82
83     #sql7="""select ReferenceID from paper_reference where PaperID='{0}'""".format(
84                                     oneid)
85
86     #cursor.execute(sql7)
87     #result5= cursor.fetchone()
88     #if result5!= None:
89     #     afforcom=list(result5)
90     #else:
91     #     afforcom=[]
92     #print(afforcom)
93     couforaff=0
94     #for i in range(0,len(allaff)):
95     #     if allaff[i] in afforcom:
96     #         couforaff+=1
97
98     if oneid in allaff:
99         couforaff+=1
100
101     #sql8="""select Title from papers where PaperID='{0}' """ .format(oneid[0])
102     #cursor.execute(sql8)
103     #result8= cursor.fetchone()
104     wordforcom=p2t[oneid]
105     #print(wordforcom)
106     couforword=0
107     for i in range(0,len(words)):
108         if words[i] in wordforcom :
109             couforword+=1
110
111     ans[oneid]=couforauthor*2+couforaff*5+couforword*15
112     j+=1
113
114     if couforauthor*2+couforaff*5+couforword*15>22:
115         k+=1
116     if k==4:
117         #print(i)
118         break

```

In this part of codes, you can see a lot of annotation, these annotation show the way to solve this problem by connecting MySQL. Of course, I abandoned this way.

In order to condense the running time further, I choose to score for each papers. I have set a standard for each indicator: if these papers have common authors, we give two grades for one commom author, and three grades

for being referenced by the paper from php, but 15 grades for each common words. Because we see it highly if two papers have enough common words. I have set an expectation as 42 grades, that means if one paper is recommended, it should at least have three words in common or two words in common and common authors or being referenced. Another way I find to condense running time is to break in time. Because we only need to show 4 papers, so the moment we find 4 papers with grades higher than our expectation, we can stop pur searching.

Now we have built a dictionary named ans, with paper id corresponding to its grades, we can sort it and get its first four results int newans.

```
122 newans=sorted(ans.items(),key=lambda d:d[1],reverse = True)
```

Now we only need to put results into a dictionary with paper id corresponding to its title. In this part, we connect to MySQL again.

```
123 forphp={}
124 b=0
125 #print(newans)
126 for items in newans:
127
128     if items[1]==len(allauthor)*2+len(allaff)*5+len(words)*15:
129         continue
130     else:
131         sql8="""select Title from papers where PaperID='{ }' """.format(items[0])
132         cursor.execute(sql8)
133         result8= cursor.fetchone()
134         #print(result8)
135         forphp[items[0]]=result8[0]
136         b+=1
137         if b==3:
138             break
```

Now we only need to post for php back to php, we only need to make use of json\_str to change it into json form, by printing, we can send it back to paper.php.

```
149 json_str = json.dumps(forphp);  
150 print(json_str);
```

Then in php, by dncoding, we can get the dictionary into an array.

decoding

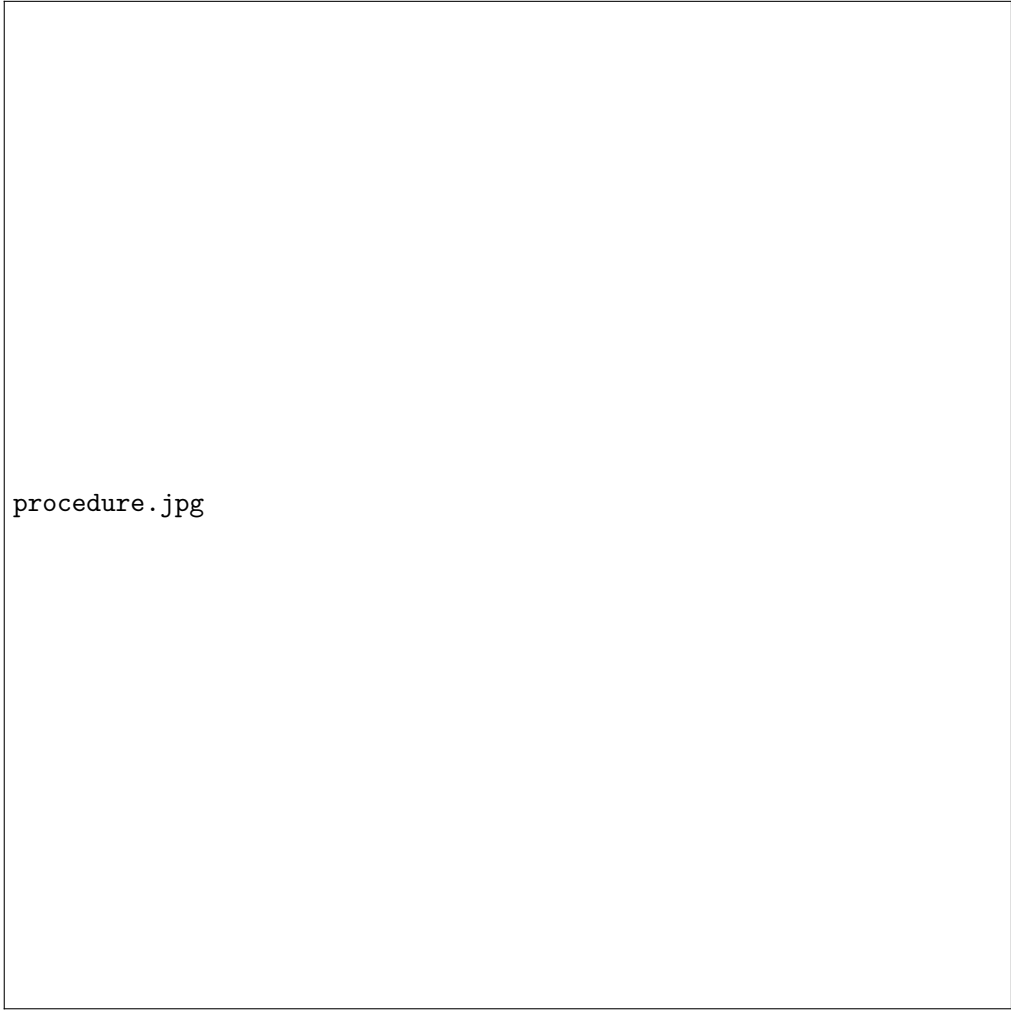
```
1 $data=json_decode($output[0], true);
```

Now, I have finished all my job.

### 3 Teachers and Students Relationship Trees

Done by Wen Tianlin.

In order to present the result of prediction of teachers and students relationship, we consider graphes are good choices. Considering conditions that one student may have several teachers and one teacher often teacher more than one student, we are determined to create two trees to present the whole result vividly. One is called the teachers tree and the other is called the students tree. Each respectively shows the relavent relationship in an order.



`procedure.jpg`

### **3.1 Backups**

We get help from ECharts to present the graphes on the website. Firstly, we installed packages provided by ECharts and added them to our website. Then on their website, they provide us with certain codes to present a graph relatively. But there are some problems before we use the code in our website.



## 3.2 Codes

The code provided by ECharts website is like the follows:

### Code of origin ECharts

```
1    myChart.showLoading();
2
3    var data1 = {
4      "name": "flare",
5      "children": [
6        {
7          "name": "data",
8          "children": [
9            {
10             ...
11            ]
12          }
13         ...;
14       var data2 = {
15         "name": "flare",
16         "children": [
17           {
18             ...
19           ]
20         };
21       myChart.hideLoading();
```

### 3.2.1 Problems of origin codes

However convenient it might seem to be to use ECharts, we could easily find that the trees can only be presented with certain and accurate information. To realize real-time predict and presentation, we need to make some improvement to the codes.

### 3.3 Improvements to the codes

Firstly, we need to collect information needed in the chart. And complete prediction when we enter the page of each scholar. Also the data need to be reglated into required form to be correctly used by ECharts.

#### 3.3.1 Python Code

We connected to Python to help us get results from MySQL and change their form. And we combined this function with the previously done prediction function. The main code is shown as follows:

##### Code of students finding

```
1     global clf
2     global result_a
3     global clf1
4     clf = joblib.load('SVC_rbf.pkl')#We have saved previously trained models
5     result_a=list(d1.keys())
6     def find_students(aid):#This function will give us the students' IDs
7         feature=[]
8         len1=len(result_a)
9         for y in range(0,len1):
10            bid=result_a[y]
11            if bid==aid:
12                continue
13            feature_array=[0,0,0,0,0,0,0,0,0,0,0]
14            a_paper=d1[aid]
15            a_paper_year=[]
16            for i in range(0,len(a_paper)):
17                a_paper_year.append((a_paper[i],d2[a_paper[i]]))
18            asort(a_paper_year)
19            b_paper=d1[bid]
20            b_paper_year=[]
21            for i in range(0,len(b_paper)):
22                b_paper_year.append((b_paper[i],d2[b_paper[i]]))
```

```

23     asort(b_paper_year)
24     c_p=[]
25     for i in range(0,len(a_paper_year)):
26         for j in range(0,len(b_paper_year)):
27             if a_paper_year[i][0]==b_paper_year[j][0]:
28                 c_p.append((i,j))
29                 break
30         if c_p:
31             for i in range(0,len(a_paper_year)):
32                 if a_paper_year[i][1]<a_paper_year[c_p[0][0]][1]:
33                     feature_array[0]+=1;
34                 else:
35                     break
36             for j in range(0,len(b_paper_year)):
37                 if b_paper_year[j][1]<b_paper_year[c_p[0][1]][1]:
38                     feature_array[1]+=1
39                 else:
40                     break
41             feature_array[2]=(feature_array[0]-feature_array[1])/len(c_p)
42             feature_array[3]=-a_paper_year[0][1]+a_paper_year[c_p[0][0]][1]
43             feature_array[4]=-b_paper_year[0][1]+b_paper_year[c_p[0][1]][1]
44             copt=-a_paper_year[c_p[0][0]][1]+a_paper_year[c_p[-1][0]][1]+1
45             feature_array[5]=(feature_array[3]-feature_array[4])/copt
46             for i in range(c_p[0][0],c_p[-1][0]):
47                 for j in range(0,len(b_paper_year)):
48                     if a_paper_year[i]==b_paper_year[j]:
49                         break
50                 else:
51                     feature_array[6]+=1
52             for i in range(c_p[0][1],c_p[-1][1]):
53                 for j in range(0,len(a_paper_year)):
54                     if b_paper_year[i]==a_paper_year[j]:
55                         break

```

```

56     else:
57         feature_array[7]+=1
58         feature_array[8]=(feature_array[6]-feature_array[7])/len(c_p)
59         feature_array[9]=len(c_p)/len(a_paper_year)
60         feature_array[10]=len(c_p)/len(b_paper_year)
61         feature.append(feature_array)
62         students=[]
63         if not feature:
64             return students
65         result=clf.predict(feature)
66         len2=len(result)
67         for i in range(0,len2):
68             if result[i]==1:
69                 students.append(result_a[i])
70         return students

```

#### Code of teachers finding

```

1     def find_teachers (bid):# This function will give us the teachers' IDs
2         feature=[]
3         len1=len(result_a)
4         for y in range(0,len1):
5             aid=result_a[y]
6             if aid==bid:
7                 continue
8             feature_array=[0,0,0,0,0,0,0,0,0,0,0]
9             a_paper=d1[aid]
10            a_paper_year=[]
11            for i in range(0,len(a_paper)):
12                a_paper_year.append((a_paper[i],d2[a_paper[i]]))
13            asort(a_paper_year)
14            b_paper=d1[bid]
15            b_paper_year=[]
16            for i in range(0,len(b_paper)):

```

```

17     b_paper_year.append((b_paper[i], d2[b_paper[i]]))
18     asort(b_paper_year)
19     c_p=[]
20     for i in range(0, len(a_paper_year)):
21         for j in range(0, len(b_paper_year)):
22             if a_paper_year[i][0]==b_paper_year[j][0]:
23                 c_p.append((i, j))
24                 break
25             if c_p:
26                 for i in range(0, len(a_paper_year)):
27                     if a_paper_year[i][1] < a_paper_year[c_p[0][0]][1]:
28                         feature_array[0]+=1;
29                     else:
30                         break
31                 for j in range(0, len(b_paper_year)):
32                     if b_paper_year[j][1] < b_paper_year[c_p[0][1]][1]:
33                         feature_array[1]+=1
34                     else:
35                         break
36                 feature_array[2]=(feature_array[0]-feature_array[1])/len(c_p)
37                 feature_array[3]=-a_paper_year[0][1]+a_paper_year[c_p[0][0]][1]
38                 feature_array[4]=-b_paper_year[0][1]+b_paper_year[c_p[0][1]][1]
39                 copt=-a_paper_year[c_p[0][0]][1]+a_paper_year[c_p[-1][0]][1]+1
40                 feature_array[5]=(feature_array[3]-feature_array[4])/copt
41                 for i in range(c_p[0][0], c_p[-1][0]):
42                     for j in range(0, len(b_paper_year)):
43                         if a_paper_year[i]==b_paper_year[j]:
44                             break
45                         else:
46                             feature_array[6]+=1
47                 for i in range(c_p[0][1], c_p[-1][1]):
48                     for j in range(0, len(a_paper_year)):
49                         if b_paper_year[i]==a_paper_year[j]:

```

```

50     break
51     else :
52         feature_array [7] += 1
53         feature_array [8] = ( feature_array [6] - feature_array [7]) / len ( c_p )
54         feature_array [9] = len ( c_p ) / len ( a_paper_year )
55         feature_array [10] = len ( c_p ) / len ( b_paper_year )
56         feature . append ( feature_array )
57         teachers = []
58         if not feature :
59             return teachers
60         result = clf . predict ( feature )
61         len2 = len ( result )
62         for i in range ( 0 , len2 ) :
63             if result [ i ] == 1 :
64                 teachers . append ( result_a [ i ] )
65         return teachers

```

#### Code of dealing results

```

1     studentsid = find_students ( idget )
2     teachersid = find_teachers ( idget )
3     lens = len ( studentsid )
4     lent = len ( teachersid )
5     students = []
6     teachers = []
7     db = pymysql . connect ( " localhost " , " root " , " " , " main_db " , charset = " utf8 " ,
8     # connect to MySQL to get names according to IDs
9     # connect to database
10    # use cursor () to create cursor
11    cursor = db . cursor ()
12    for i in studentsid :
13        a_sql = " " " select authors . AuthorName from authors where AuthorID = ' { } ' " " " . f
14        cursor . execute ( a_sql )
15        result_a = cursor . fetchall ()

```

```

16     students.append(result_a[0][0][: -1])
17     for j in teachersid:
18         b_sql="""select authors.AuthorName from authors where AuthorID={ }""".f
19         cursor.execute(b_sql)
20         result_b=cursor.fetchall()
21         teachers.append(result_b[0][0][: -1])
22         if lens<lent:
23             for i in range(0,lent-lens):
24                 students.append(str(i))
25             if lens>lent:
26                 for i in range(0,lens-lent):
27                     teachers.append(str(i))
28
29         res={}
30         for i in range(0,max(lent , lens )):
31             res [ teachers [ i ]]=students [ i ]
32         forphp=json.dumps(res)
33         print(forphp)

```

### 3.3.2 PHP code

We called the python file we had designed and sent target authorID to conduct the research, and prepared arrays to hold the results in order to be used in the later procedures.

#### Code of PHP part

```

1     <?php
2     exec('PythonApplication19.py ' . $author,$output);
3     $array1 = json_decode($output[0],true);
4     $teachers=array();
5     $students=array();
6     foreach ($array1 as $key=>$value) {
7         if (strlen($key)>2)
8         {array_push($teachers,$key);}

```

```

9     if (strlen($value)>2)
10    {array_push($students,$value);echo $value;}
11    }
12    $stunum=count($students);
13    $teanum=count($teachers);
14    $i=0;
15    for ($i=0;$i<$stunum;$i+=1)
16    {echo $students[$i];}
17    ?>

```

### 3.3.3 Java Script code

Also, ECharts needed Java Script to call its package and make settings for it, including prepare a certain-size dom and call the arrays created.

#### Code of Java Script part

```

1     <!-- prepare a certain size dom for ECharts -->
2     <div id="main" style="width:800px;height:600px;"></div>
3     <script type="text/javascript">
4     // Initialize the ECharts
5     var myChart = echarts.init(document.getElementById('main'));
6
7     // Conduct Settings and Data
8
9     // Use the upper elements to show the chart.
10
11    myChart.showLoading();
12    var teachernum=<?php echo $teanum?>;
13    var teacherarray=new Array();
14    var i;
15    <?php for ($i=0;$i<$teanum;$i++)
16    {
17    ?>
18    teacherarray[<?php echo $i;?>]={"name": "<?php echo $teachers[$i]?>", "va

```



```

19     <?php }?>
20     var data1 = {
21         "name": "<?php echo $author_name;?>",
22         "children": teacherarray ,
23     };
24
25     var studentnum=<?php echo $stunum?>;
26     var studentarray=new Array ();
27     <?php for ($i=0;$i<$stunum;$i++)
28     {
29     ?>
30     studentarray[<?php echo $i;?>]={"name": "<?php echo $students [$i]?>", "va
31     <?php }?>
32     var data2 = {
33         "name": "<?php echo $author_name;?>",
34         "children": studentarray ,
35     };

```

### 3.3.4 ECharts code

The last step was to put all those things back to ECharts code and thus we would get the charts.

#### Code of final Echarts

```

1     myChart.hideLoading ();
2     myChart.setOption(option = {
3         tooltip: {
4             trigger: 'item',
5             triggerOn: 'mousemove'
6         },
7         legend: {
8             top: '2%',
9             left: '3%',
10            orient: 'vertical',

```

```

11     data: [{
12     name: 'Possible Teachers Tree',
13     icon: 'rectangle'
14     } ,
15     {
16     name: 'Possible Students Tree',
17     icon: 'rectangle'
18     }],
19     borderColor: '#c23531'
20     },
21     series:[
22     {
23     type: 'tree',
24     name: 'Possible Teachers Tree',
25     data: [data1]
26     top: '5%',
27     left: '20%',
28     bottom: '2%',
29     right: '70%',
30     symbolSize: 7,
31     label: {
32     normal: {
33     position: 'left',
34     verticalAlign: 'middle',
35     align: 'right'
36     }
37     },
38     leaves: {
39     label: {
40     normal: {
41     position: 'right',
42     verticalAlign: 'middle',
43     align: 'left'

```

```

44     }
45     }
46   },
47   expandAndCollapse: true ,
48   animationDuration: 550,
49   animationDurationUpdate: 750
50   },
51   {
52     type: 'tree ',
53     name: 'Possible Students Tree ',
54     data: [data2],
55     top: '5%',
56     left: '70%',
57     bottom: '2%',
58     right: '20%',
59
60     symbolSize: 7,
61
62     label: {
63       normal: {
64         position: 'left ',
65         verticalAlign: 'middle ',
66         align: 'right '
67       }
68     },
69
70     leaves: {
71       label: {
72         normal: {
73           position: 'right ',
74           verticalAlign: 'middle ',
75           align: 'left '
76       }

```

```
77     }
78     },
79     expandAndCollapse: true ,
80     animationDuration: 550,
81     animationDurationUpdate: 750
82     }
83     ]
84     });
85 </script >
```

### 3.4 Result Tree



### 3.5 Possible Improvement

#### 3.5.1 Running Time

Now the whole procedure takes much time to get results and present it on the page. This is due to too much calling between different software and elements. Maybe calling MySQL from PHP directly might help.

### **3.5.2 Deeper Trees**

All trees are presenting only one storey of children now. We previously aim to add click events to the tree to enable it present the second storey in real-time.

## **4 Beautification of the Webpage**

Done by GU Shixuan.

Besides the powerful functions, to make a nice webpage, a rather beautiful appearance is a must simply because it can add to the attractiveness to the users, thus bringing better experiences.

In this way, I made some embellishments and added some other elements to the project.

## 4.1 PROCEDURE

The embellishments can mainly be divided into three parts—the dynamic background, the navigation, the type setting and some modifications of typeface.

### 4.1.1 The dynamic background designing.

The background i designed was a canvas with many particles generally emerging, connected to each other, which actually symbolizes the rather close relation between different subjects. So it also acts in cooperation with the title ‘ACADEMIC SEARCH’.

In terms of the realization of the background, it can be expressed as following:

I first created the canvas and a particle class, which later I could use to display the particles on the canvas. And since I wanted to make it a dynamic effect, I used random-generated funtions so that the particles could emerge out of order and their color is also randomly selected. In this way, I also made the size and appearance of the particles randomly emerged by making the radius and type elements the random variable.

Code1

```
1 var tela = document.createElement('canvas');
2 tela.width = $(window).width();
3 tela.height = $(window).height();
4 $("body").append(tela);
5
6 var canvas = tela.getContext('2d');
7
8 class Particle{
9 constructor(canvas, options){
```

```

10 let colors = ["#feca00", "#a9df85", "#5dc0ad", "#ff9a00", "#fa3f20"]
11 let types = ["full", "fill", "empty"]
12 this.random = Math.random()
13 this.canvas = canvas;
14 this.progress = 0;
15
16 this.x = ($(window).width()/2) + (Math.random()*200 - Math.random()
17 *200)
18 this.y = ($(window).height()/2) + (Math.random()*200 - Math.random()
19 *200)
20 this.w = $(window).width()
21 this.h = $(window).height()
22 this.radius = 1 + (8*this.random)
23 this.type = types[this.randomIntFromInterval(0, types.length - 1)];
24 this.color = colors[this.randomIntFromInterval(0, colors.length - 1)];
25 this.a = 0
26 this.s = (this.radius + (Math.random() * 1))/10;
27 //this.s = 12 //Math.random() * 1;
28 }
29
30 getCoordinates(){
31 return {
32 x: this.x,
33 y: this.y
34 }
35 }

```

What's worth mentioning is that the realizing functions were kept in the js file and were imported in the home page. And I set a upper limitation of the maximum number of particles to make sure that it wouldn't fill in the whole page. To make the particles move, I had to create their arclengths. I created two functions to generate acrlength on the canvas and because the particles' parameters were randomly created, I took the advantage of it to created their different acrlengths.



## Code2

```
1  randomIntFromInterval(min,max){
2  return Math.floor(Math.random()*(max-min+1)+min);
3  }
4
5  render(){
6  // Create arc
7  let lineWidth = 0.2 + (2.8*this.random);
8  let color = this.color;
9  switch(this.type){
10 case "full":
11   this.createArcFill(this.radius, color)
12   this.createArcEmpty(this.radius+lineWidth, lineWidth/2, color)
13   break;
14 case "fill":
15   this.createArcFill(this.radius, color)
16   break;
17 case "empty":
18   this.createArcEmpty(this.radius, lineWidth, color)
19   break;
20 }
21 }
22
23 createArcFill(radius, color){
24   this.canvas.beginPath();
25   this.canvas.arc(this.x, this.y, radius, 0, 2 * Math.PI);
26   this.canvas.fillStyle = color;
27   this.canvas.fill();
28   this.canvas.closePath();
29 }
30
31 createArcEmpty(radius, lineWidth, color){
32   this.canvas.beginPath();
```

```

33 this.canvas.arc(this.x, this.y, radius, 0, 2 * Math.PI);
34 this.canvas.lineWidth = lineWidth;
35 this.canvas.strokeStyle = color;
36 this.canvas.stroke();
37 this.canvas.closePath();
38 }
39
40 move(){
41
42 this.x += Math.cos(this.a) * this.s;
43 this.y += Math.sin(this.a) * this.s;
44 this.a += Math.random() * 0.4 - 0.2;
45
46 if(this.x < 0 || this.x > this.w - this.radius){
47 return false
48 }
49
50 if(this.y < 0 || this.y > this.h - this.radius){
51 return false
52 }
53 this.render()
54 return true
55 }

```

The line between two particles was rather an easy task, by adding the parameters to the following codes, it could be simply finished. The last step was to set a boundary which means once the particle approaches the boundary, it will be deleted and a new particle will emerge. It can be deleted by changing the parameter of fillrect function into 0,0, thus it will be out of the screen.

### Code3

```

1 calculateDistance(v1, v2){
2 let x = Math.abs(v1.x - v2.x);

```

```

3 let y = Math.abs(v1.y - v2.y);
4 return Math.sqrt((x * x) + (y * y));
5 }
6 }
7
8
9 function populate(num){
10 for (var i = 0; i < num; i++) {
11 setTimeout(
12 function(x){
13 return function(){
14 particles.push(new Particle(canvas))
15 };
16 }(i)
17 ,frequency*i);
18 }
19 return particles.length
20 }
21
22 function clear(){
23 canvas.fillStyle='#111111';
24 canvas.fillRect(0, 0, tela.width, tela.height);
25 }
26
27 function connection(){
28 let old_element = null
29 $.each(particles, function(i, element){
30 if(i>0){
31 let box1 = old_element.getCoordinates()
32 let box2 = element.getCoordinates()
33 canvas.beginPath();
34 canvas.moveTo(box1.x, box1.y);
35 canvas.lineTo(box2.x, box2.y);

```

```

36 canvas.lineWidth = 0.45;
37 canvas.strokeStyle="#3f47ff";
38 canvas.stroke();
39 canvas.closePath();
40 }
41
42 old_element = element
43 })
44 }
45
46
47 function update(){
48 clear();
49 connection()
50 particles = particles.filter(function(p) { return p.move() })
51 if(time_to_recreate){
52 if(particles.length < init_num){ populate(1);}
53 }
54 requestAnimationFrame(update.bind(this))
55 }

```

#### 4.1.2 The navigation designing.

Since the project asked for three different searching modes in our homepage, I designed the relevant navigation on the homepage which realized the function of guiding the users to the responding pages.

In terms of the realization part, I simply created the 'li' elements with hyperlinks in the body part. And I made some modifications in the head part as well. For instance, I added some hover effects to them and altered the type settings.

Code4

```
1 <ul>
```

```

2 <li><a href="/home.php">Home</a></li>
3 <li><a href="/home1.php">Conferences</a></li>
4 <li><a href="/home2.php">Papers</a></li>
5 <li><a href="http://ieeeseiee.com/>Contact</a></li>
6 <li><a href="/about.php">About US</a></li>
7 </ul>

```

### 4.1.3 Modifications of typeface

In this aspect, I rearranged the page by changing the positions of the elements. The complete block was put in the center of the home page and the input box's text color was changed into white considering the dark background. And I also put the navigation part in the upper part of the page.

#### Code5

```

1 <style>
2 .more-pens {
3   position: fixed;
4   left: 20px;
5   bottom: 20px;
6   z-index: 10;
7   font-family: "Montserrat";
8   font-size: 12px;
9 }
10
11 a.white-mode, a.white-mode:link, a.white-mode:visited,
12 a.white-mode:active {
13   font-family: "Montserrat";
14   font-size: 12px;
15   text-decoration: none;
16   background: #212121;
17   padding: 4px 8px;

```

```
18 color: #f7f7f7;
19 }
20 a.white-mode:hover, a.white-mode:link:hover,
21 a.white-mode:visited:hover, a.white-mode:active:hover {
22 background: #edf3f8;
23 color: #212121;
24 }
25
26 body {
27 margin: 0;
28 padding: 0;
29 overflow: hidden;
30 width: 100%;
31 height: 100%;
32 background: #000000;
33 }
34
35 .title {
36 z-index: 10;
37 position: absolute;
38 left: 50%;
39 top: 30%;
40 transform: translateX(-50%) translateY(-50%);
41 font-family: "Montserrat";
42 width: 100%;
43 }
44 .title h1 {
45 position: relative;
46 color: #EEEEEE;
47 font-weight: 600;
48 font-size: 60px;
49 padding: 0;
50 margin: 0;
```

```
51 line-height: 1;
52 text-shadow: 0 0 30px #000155;
53 text-align: center;
54 }
55 .title h1 span {
56 font-weight: 600;
57 padding: 0;
58 margin: 0;
59 color: #BBB;
60 text-align: center;
61 }
62 .title h3 {
63 text-align: center;
64 font-weight: 200;
65 font-size: 20px;
66 padding: 0;
67 margin: 0;
68 line-height: 1;
69 color: #EEEEEE;
70 letter-spacing: 2px;
71 text-shadow: 0 0 30px #000155;
72 }
73 * {
74 box-sizing: border-box;
75 }
76
77 form
78 { text-align: center;
79 left: 20%; bottom: 30%;
80 }
81
82 input[type=text] {
83
```

```

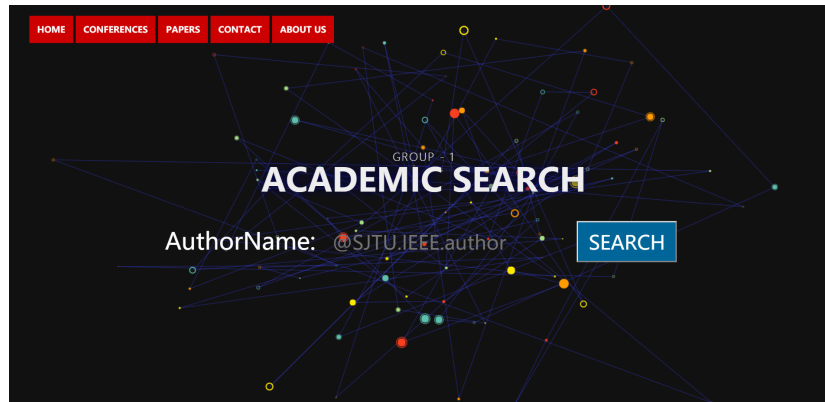
84 width:30%;
85 padding:12px 20px;
86 }
87 input [type=submit] {
88
89 width:12%;
90 padding:12px 20px;
91 }
92 input [type=submit] {
93
94 background-color:#006699;
95 color:white;
96 }
97 input [type=submit]:hover {
98
99 background-color:#000066;
100 }
101
102 input {color:#fff;}
103 <!--
104 body,td,th {
105 color: #000000;
106 }
107 .STYLE1 {
108 font-family: "weiruanyahei";
109 font-size: 36px;
110 }
111
112 ul
113 {
114 list-style-type:none;
115 margin:0;
116 padding:0;

```



```
117 padding-top:25px;
118 padding-bottom:200px;
119 }
120 li
121 {
122 display:inline;
123 }
124 a:link ,a:visited
125 {
126 font-weight:bold;
127 color:#FFFFFF;
128 background-color:#CC0000 ;
129 text-align:center;
130 padding:15px;
131 text-decoration:none;
132 text-transform:uppercase;
133 }
134 a:hover ,a:active
135 {
136 background-color:#7A991A;
137 }
138 .STYLE3 {
139 color: #FFFFFF;
140 font-family: "weiruanyabei";
141 }
142 —>
143 </style>
```

## 4.2 THE RESULT DISPLAY



## 5 REFERENCE

No references.