

CS235 Data Mining Techniques

Assignment Report

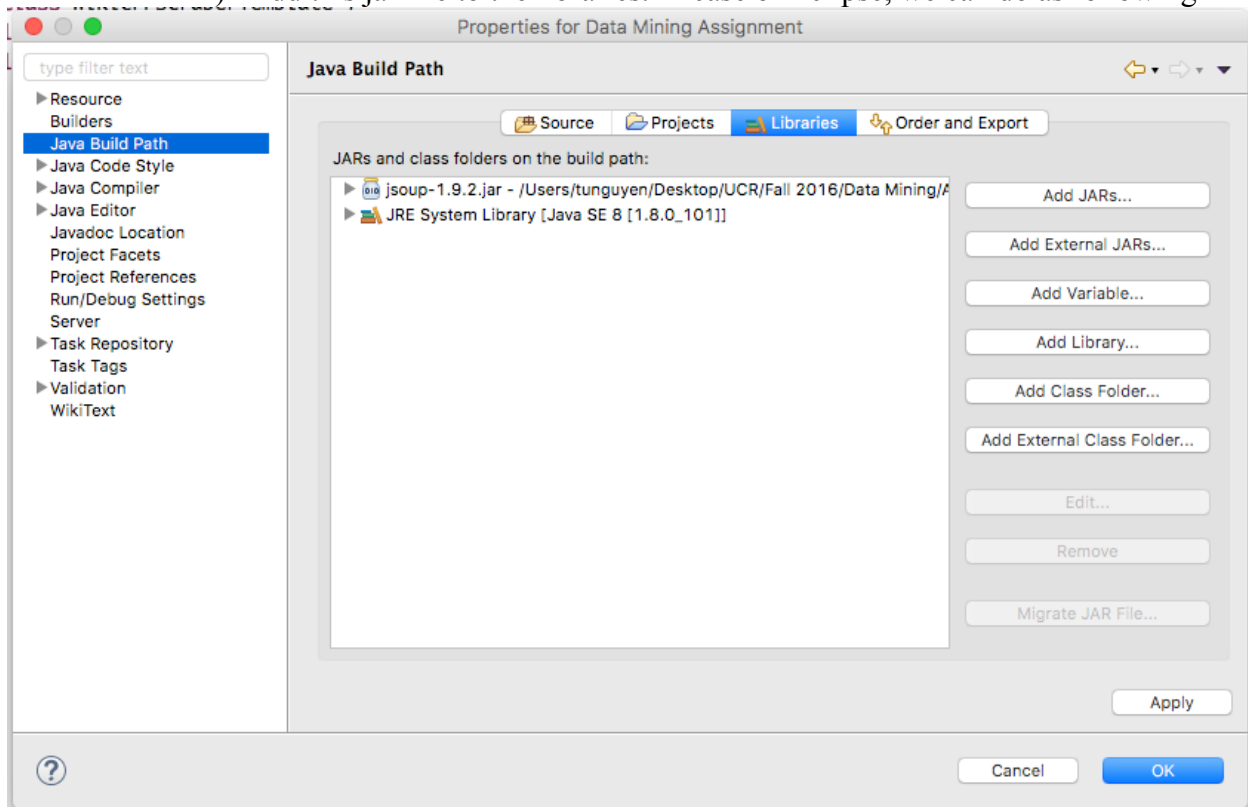
November 12, 2016

Student Name **Student ID**
Tu Nguyen 861309226

I. Data Crawling

I used Jsoup library (<https://jsoup.org/>) to parse HTML pages. Here are the steps to use Jsoup

- 1) Download the jar file (jsoup-1.9.2.jar) and add it to the location of the project file
- 2) Add this jar file to the libraries. In case of Eclipse, we can do as following



- 3) Import the library so that we can use its APIs

```
import org.jsoup.Jsoup;  
import org.jsoup.nodes.Document;  
import org.jsoup.nodes.Element;  
import org.jsoup.select.Elements;
```

I crawled all pages of all conferences and saved them to one file "wikicfp_crawl.txt".

```
String category[] = {"data mining", "databases", "machine learning",  
"artificial intelligence"};
```

```

        //String category = "databases";
        //String category = "machine learning";
        //String category = "artificial intelligence";

        int numOfPages = 20;

        //create the output file
        File file = new File("wikicfp_crawl.txt");
        file.createNewFile();
        FileWriter writer = new FileWriter(file);

```

The output file has 5 columns separated by tab as following

```

        /* Write columns' headers to the target file */
        writer.write("Conference's Acronym with year\t");
        writer.write("Conference's Acronym\t");
        writer.write("Year\t");
        writer.write("Conference's Name\t");
        writer.write("Location\n");

```

Here is the main code to parse the HTML pages, extract the necessary information, and save them to the output file

```

        //now start crawling the all 'numOfPages' pages
        for (int categoryIndex = 0; categoryIndex <
category.length; categoryIndex++) {
            for(int i = 1;i<=numOfPages;i++) {
                //Create the initial request to read the first
page
                //and get the number of total results
                String linkToScrape =
"http://www.wikicfp.com/cfp/call?conference="+
URLLEncoder.encode(category[categoryIndex], "UTF-8")+"&page=" + i;
                String content = getPageFromUrl(linkToScrape);

                Document doc = Jsoup.parse(content);

                /* Navigate the content section */
                Element cotentSec =
doc.select("div.contsec").first();

                /* Select the first table, which contains
necessary info of conferences */
                Element table =
cotentSec.select("table").get(0);

```

```

        Elements rows = table.select("tr");

        for (int j = 0; j < rows.size(); j++) {
            Element row = rows.get(j);
            Elements cols = row.select("td");

            /* Search for column containing the key
word "Event" */

            if (cols.get(0).text().contains("Event")) {
                /* Target content identified */

                /* The first table contains all info of
conferences */

                Element tTable =
cols.get(0).select("table").first();
                Elements tRows = tTable.select("tr");
                int numRows = tRows.size();

                /* Skip the first row, which contains
columns' titles */

                int k = 1;
                while (k < numRows) {
                    /* Each conference record has two
consecutive rows:
conference name
deadline */

                    Element firstRow, secondRow;

                    firstRow = tRows.get(k++);
                    if (k < numRows) {
                        Elements firstRowCols;
                        firstRowCols =
firstRow.select("td");

                        if (firstRowCols.size() == 1) {
                            /* Skip this row since it
does not contain necessary info */

                            System.out.println(firstRowCols.get(0).text());
                        }
                        else {
                            Elements secondRowCols;

```

```

/* We want to search for
pattern of "Conference's acronym + space or ' + conference's year" */
Pattern p =
Pattern.compile("(.)[\\s\\'](\\d+)");
Matcher m;

/* Get the second row of
a record */
secondRow =
tRows.get(k++);
secondRowCols =
secondRow.select("td");

/* Write conference's
acronym with year to the target file */
writer.write(firstRowCols.get(0).text() + "\t");

/* Get conference's
acronym and year */
m =
p.matcher(firstRowCols.get(0).text());
//m =
p.matcher("ICNP2016");
if (m.find()) {
    /* Write the
conference's acronym to the target file */
    writer.write(m.group(1) + "\t");
    //System.out.println(m.group(1));

    /* Write the
conference's year to the target file */
    writer.write(m.group(2) + "\t");
    //System.out.println(m.group(2));
}
else {
    /* We need a new
pattern for this special case */
    Pattern p1 =

```

```

Pattern.compile("(\\D+)(\\d+)");
p1.matcher(firstRowCols.get(0).text());

conference's acronym to the target file */
        writer.write(m1.group(1) + "\\t");
        //System.out.println(m1.group(1));

conference's year to the target file */
        writer.write(m1.group(2) + "\\t");
        //System.out.println(m1.group(2));

whatever we have for acronym to the target file */
        writer.write(firstRowCols.get(0).text() + "\\t");

be NA */
        writer.write("NA" + "\\t");
        System.out.println("Year is NA");

name to the target file */
        writer.write(firstRowCols.get(1).text() + "\\t");

location to the target file */
        writer.write(secondRowCols.get(1).text() + "\\n");

```

```

    }
    else {
        /* Invalid record */
        System.out.println("Error on
page #" + i + ", k = " + k);
        break;
    }
}

/* Already got all necessary info */
break;
}
}

//System.out.println(content);

//IMPORTANT! Do not change the following:
Thread.sleep(DELAY*1000); //rate-limit the
queries
    }
}

writer.close();
} catch (IOException e) {
    e.printStackTrace();
} catch (InterruptedException e) {
    e.printStackTrace();
}
}
}

```

II. Data Cleaning

I used OpenRefine to clean the data. Here are the steps:

1. Capitalize and remove spaces at the beginning and end of the column "Conference's acronym with year"
2. Do "Text facet" on the column "Conference's acronym with year" ("No spaces")

1600 rows						
Show as: rows records Show: 5 10 25 50 rows						
All	No Spaces	Conference's Acronym	Year	Conference's Name	Location	
1. Facet	Text facet		2016	Second International Conference on Data Mining and Applications	Vienna, Austria	
2. Text filter	Numeric facet		2016	Integrated Optics and Lightwave: An International Journal	N/A	
3. Edit cells	Timeline facet		2016	Second International Conference on Advances in Computer Science and Information Technology	Chennai	
4. Edit column	Scatterplot facet		2016	International Journal on Organic Electronics	N/A	
5. Transpose	Custom text facet...		2016	International Journal of Artificial Intelligence & Applications	N/A	
6. Sort...	Custom Numeric Facet...		2016	International Journal of Antennas	N/A	
7. View	Customized facets		2016	International Journal of Database Management Systems	N/A	
8. Reconcile			2016	International Journal of Data Mining & Knowledge Management Process	N/A	
9. ICNCC - ACM 2016	ICNCC - ACM	2016	2016	International Journal of Computer Science and Information Technology	N/A	
10. L@S 2017	L@S	2017	2016	International Journal of Computer Science, Engineering and Applications	N/A	
11. L@S 2017	L@S	2017	2016	5th International Conference on Network, Communication and Computing - ACM	Kyoto, Japan	
12. L@S 2017	L@S	2017	2016	L@S 2017: Fourth Annual ACM Conference on Learning at Scale	Massachusetts	
13. L@S 2017	L@S	2017	2016	International Journal of Education	N/A	
14. SPTM 2016	SPTM	2016	2016	Fourth International Conference of Security, Privacy and Trust Management	Chennai	
15. EEIEJ 2016	EEIEJ	2016	2016	Emerging Trends in Electrical, Electronics & Instrumentation Engineering: An International Journal	N/A	
16. BAKTH 2017	BAKTH	2017	2016	The 21st Pacific Asia Conference on Knowledge Discovery and Data Mining	India: Hyderabad	

- Choose “Sort by: count”
- Manually browse each groups to see if they really belong to the same group. If they are different, modify their names

6 matching rows (1600 total)						
Show as: rows records Show: 5 10 25 50 rows						
All	No Spaces	Conference's Acronym	Year	Conference's Name	Location	
112. AI 2016 Joint Conference	AI Joint Conference	2016	2016	The 29th Australasian Joint Conference on Artificial Intelligence	Hobart, Australia	
218. AI 2016	AI	2016	2016	29th Canadian Conference on Artificial Intelligence	Victoria, BC, Canada	
912. AI 2016 Joint Conference	AI Joint Conference	2016	2016	The 29th Australasian Joint Conference on Artificial Intelligence	Hobart, Australia	
1066. AI 2016	AI	2016	2016	29th Canadian Conference on Artificial Intelligence	Victoria, BC, Canada	
1363. AI 2016 Joint Conference	AI Joint Conference	2016	2016	The 29th Australasian Joint Conference on Artificial Intelligence	Hobart, Australia	
1588. AI 2016	AI	2016	2016	29th Canadian Conference on Artificial Intelligence	Victoria, BC, Canada	

- Remove duplicate rows by doing the following steps
 - Sort the column “Conference’s acronym with year” (“No spaces”)

Sort by No Spaces

Sort cell values as

☒ text
 ☐ case-sensitive

☐ numbers

☐ dates

☐ booleans

Position blanks and errors

Valid values

Errors

Blanks

Drag and drop to re-order

☒ a - z
 ☐ z - a

OK Cancel

b. Choose “Sort -> Reorder rows permanently”

1600 rows

Show as: **rows** records Show: 5 10 25 50 rows Sort ▾

▼ All	▼ No Spaces		▼ Year
★	1184.	(***CLOSED***) CALL FOR BOOK CHAPTERS 201	2015
★	235.	(BDM'16) 2016	2016
★	1308.	[ICCIS] 2016	2016
★	1305.	[ICMDM] 2016	2016
★	1115.	4DML IN BIOMETRICS 2015	2015
★	1449.	6TH ICITCS 2016	2016

Remove sort

Reorder rows permanently

By No Spaces ▶

c. Edit cells -> Blank down

1600 rows				
Show as: rows records		Show: 5 10 25 50 rows		
▼ All	▼ No Spaces	▼ Conference's Acronym		
★	1.	Facet	OR BOOK CHAPTERS 2015	(***Closed***) Call For Book Chapters
★	2.	Text filter		(BDM'16)
★	3.	Edit cells		[ICCIS]
★	4.	Edit column		[ICMDM]
★	5.	Transpose		4DML in Biometrics
★	6.	Sort...		6th ICITCS
★	7.	View		8th ICISA
★	8.	Reconcile		AAIA
★	9.			AAMAS
★	10.			ACALCI
★	11.			ACCVW-MLAC
★	12.			ACIIDS
★	13.			ACIIDS
★	14.			ACIRS
★	15.			ACIRS

d. Facet -> Customized facet -> Facet by blank

1600 rows				
Show as: rows records		Show: 5 10 25 50 rows		
▼ All	▼ No Spaces	▼ Conference's Acronym	▼ Year	
☆ 1.	Facet ▶	(***Closed***) Call For Book Chapters	201	
☆ 2.	Text filter	(BDM'16)	201	
☆ 3.	Edit cells ▶	[ICCIS]	201	
☆ 4.	Edit column ▶	[ICMDM]	201	
☆ 5.	Transpose ▶	4DML in Biometrics	201	
☆ 6.	Sort...	6th ICITCS	201	
☆ 7.	View ▶	8th ICISA	201	
☆ 8.	Reconcile ▶	Word facet	201	
☆ 9.		Duplicates facet	201	
☆ 10.		Numeric log facet	201	
☆ 11.	ACCVW-MLAC 2016	1-bounded numeric log facet	201	
☆ 12.	ACIIDS 2012	Text length facet	201	
☆ 13.	ACIIDS 2013	Log of text length facet	201	
☆ 14.	ACIRS 2016	Unicode char-code facet	201	
☆ 15.		Facet by error	201	
☆ 16.	ACIRS 2017	Facet by blank	201	
☆ 17.	ACIRS-IEEE 2017			
☆ 18.	ACIS 2016			
☆ 19.				

e. Click on “true”

296 matching rows (1600 total)

Show as: **rows** records Show: 5 10 25 50 rows

▼ All	▼ No Spaces	▼ Conference's Acronym
Facet ▶		ACIRS
Edit rows ▶	Star rows	
Edit columns ▶	Unstar rows	
View ▶	Flag rows	
	Unflag rows	
☆	🗨	31.
☆	🗨	47.
☆	🗨	49.
☆	🗨	52.
		AECIA

- Remove the column "Conference's acronym with year" ("No spaces")
- Apply "Text facet" on the column "Conference's Acronym"
- Choose "Sort by: count"
- Remove special characters such as "-IEEE", "IEE ", "-ACM", "ACM", "(", ")", "6th", "8th", "-", ... at the beginning or end of the field since these characters will make the "Conference's Acronym" inconsistent.

Facet / Filter		Undo / Redo 21		1304 rows	
Refresh		Reset All		Show as: rows records	
Show: 5 10 25 50 rows					
Conference's Acronym		change			
1088 choices		Sort by: name count		Cluster	
BDAS 6					
VLDB 6					
COMAD 5					
DaWaK 5					
SIGMOD 5					
AMW 4					
CIKM 4					
CloudDB 4					
DBDBD 4					
DIM 4					
ICDE 4					
KDD 4					
All		Conference's Acronym		Year	
1. (**Closed**) Call For Book Chapters				2015	
2. (BDM'16)				2016	
3. [ICCIS]				2016	
4. [ICMDM]				2016	
5. 4DML in Biometrics				2015	
6. 6th ICITCS				2016	
7. 8th ICISA				2017	
8. AAIA				2016	
9. AAMAS				2017	
10. ACALCI				2016	
11. ACCVW-MLAC				2016	
12. ACIIDS				2012	
13. ACIIDS				2013	
14. ACIRS				2016	
15. ACIRS				2017	
16. ACIRS-IEEE				2017	
17. ACIS				2016	
18. ACM - DTTA Track				2013	
19. ACM-ICDLT				2017	
20. ACML				2015	
21. ACML				2016	
22. ACMME				2016	
23. ACMME-EI				2016	
24. ACM SAC-RS				2017	
25. ACSIJ v5 14 July				2016	
26. ACSTY				2016	
27. ACTION				15	
28. ACTIVE				2017	
29. ADAPT				2016	
30. Adaptive NLP				2015	
31. ADBIS				2011	
32. ADBIS -				2015	
33. ADBIS				2017	
34. ADBIS - Workshops				2017	

10. Do “Text Facet” on the column “Location”. We see that there are 83 N/A rows among 1304 rows. I think we can remove these rows since the counts for only 6.4%

Facet / Filter		Undo / Redo 21		1304 rows	
Refresh		Reset All		Show as: rows records	
Show: 5 10 25 50 rows					
Location		change			
671 choices		Sort by: name count		Cluster	
N/A 83					
Singapore 31					
Chengdu, China 17					
Hong Kong 17					
All		Conference's Acronym		Year	
1. (**Closed**) Call For Book Chapters				2015	
2. (BDM'16)				2016	
3. [ICCIS]				2016	
4. [ICMDM]				2016	
5. 4DML in Biometrics				2015	
6. 6th ICITCS				2016	
7. 8th ICISA				2017	

a) Do “Text filter” on the column “Location”, type “N/A” for the filter

Facet / Filter
Undo / Redo 21

Refresh
Reset All
Remove All

Location
change

1 choices
Sort by: name count
Cluster

N/A 83

Facet by choice counts

Location

☐ case sensitive
☐ regular expression

83 matching rows (1304 total)

Show as: rows records
Show: 5 10 25 50 rows

<input checked="" type="checkbox"/> All	<input type="checkbox"/> Conference's Acronym
<input type="checkbox"/> <input type="checkbox"/>	1. (**Closed**) Call For Book Chapters
<input type="checkbox"/> <input type="checkbox"/>	25. ACSIJ v5 i4 July
<input type="checkbox"/> <input type="checkbox"/>	62. AI Matters - Apr
<input type="checkbox"/> <input type="checkbox"/>	63. AI Matters - Mar
<input type="checkbox"/> <input type="checkbox"/>	78. AJIS (Business Analytics)
<input type="checkbox"/> <input type="checkbox"/>	90. ANTJ
<input type="checkbox"/> <input type="checkbox"/>	100. ASC-BA
<input type="checkbox"/> <input type="checkbox"/>	101. ASCE-ASME: MRRA
<input type="checkbox"/> <input type="checkbox"/>	110. AURO-DR
<input type="checkbox"/> <input type="checkbox"/>	120. BCI for Neurorobotics

b) All -> Edit rows -> Remove all matching rows

▼ All	▼ Conference's Acronym	▼ Year
Facet	losed***) Call For Book Chapters	2015
	...F... ..	2016
Edit rows	Star rows	2016
Edit columns	Unstar rows	2016
View	Flag rows	2015
	Unflag rows	2016
☆	100. ASC	2016
☆	101. ASC	2016
☆	110. AUR	2017
☆	120. BCI for Neurorobotics	2017

11. The column “Location” contains the information about city and country.
However, we just need the city information for the statistics. Thus, we need to extract the city info from this column as following: Split the column “Location” by “;”, “(”, and “-”; Keep the first column and rename the column as “City”
12. Do “Text facet” on the column “City” and manually make the city names persistent

Bangkok	9
Barcelona	23
Barcelona	1
BARCELONA Spain	1
Bari	2
Będlewo	1
Beijing	15

III. Hadoop

The input file, which contains list of conferences, has the following format

"Conference Acronym" \t "Year" \t "Conference Name" \t "Location"

1. Compute and plot the number of conferences per city
Each line in the input file contains information of one conference. Thus, the number of occurrences of a city in the file is the number of conferences of that city.

To do this computation, I just slightly modify the WordCount example as following: Use "city" name as key and number of its occurrences as value since. The output file has two columns: "City" and "Number of conferences". The map function of the Mapper looks like

```
@Override
public void map(Object key, Text value, Context context
                ) throws IOException, InterruptedException {
    String line = (caseSensitive) ?
        value.toString() : value.toString().toLowerCase();
    for (String pattern : patternsToSkip) {
        line = line.replaceAll(pattern, "");
    }
    /* Format of the input data is
     * "Conference Acronym" \t "Year" \t "Conference Name" \t "Location" */
    /* Split by tab */
    StringTokenizer itr = new StringTokenizer(line, "\t");
    /* Skip the first three columns */
    for (int i = 0; i < 3; i++) {
        if (itr.hasMoreTokens()) {
            itr.nextToken();
        } else {
            System.err.println("Bad line: " + line + "\n");
            break;
        }
    }
    /* We need to process the "City" column */
    if (itr.hasMoreTokens()) {
        city.set(itr.nextToken());
        /* Use "city" column as key and number of occurrences as value */
        context.write(city, one);
        Counter counter = context.getCounter(CountersEnum.class.getName(),
            CountersEnum.INPUT_WORDS.toString());
        counter.increment(1);
    }
}
```

The reduce function of the Reducer looks like

```
public static class IntSumReducer
    extends Reducer<Text, IntWritable, Text, IntWritable> {
    private IntWritable result = new IntWritable();

    public void reduce(Text key, Iterable<IntWritable> values,
                      Context context
                      ) throws IOException, InterruptedException {
        int sum = 0;
```



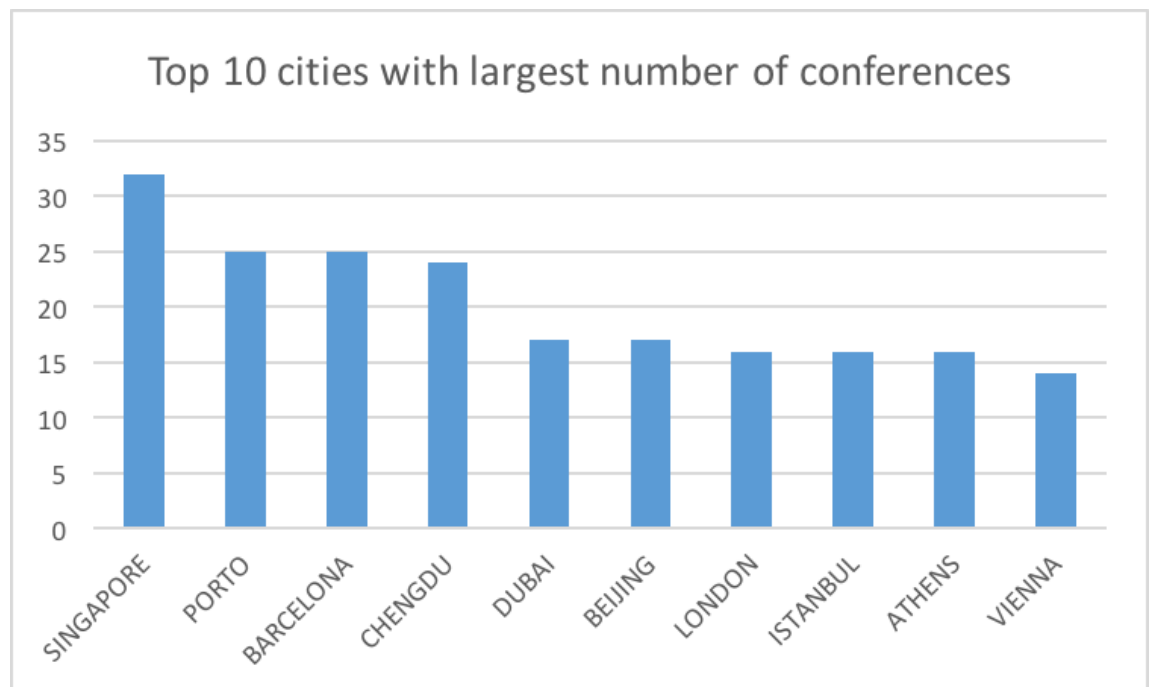
```

    for (IntWritable val : values) {
        /* Sum up number of conferences (value) */
        sum += val.get();
    }
    result.set(sum);
    context.write(key, result);
}
}

```

The top 10 locations are

City	Number of conferences
SINGAPORE	32
PORTO	25
BARCELONA	25
CHENGDU	24
DUBAI	17
BEIJING	17
LONDON	16
ISTANBUL	16
ATHENS	16
VIENNA	14



- Output the list of conferences per city

I used the “city” name (Text) as key and “conference acronym” (Text) as value.
 The output file has two columns: “City” and “Conference list”, which contains list of conferences’ acronyms separated by “,”.

The map function of the Mapper looks like

```
public void map(Object key, Text value, Context context
    ) throws IOException, InterruptedException {
    String line = (caseSensitive) ?
        value.toString() : value.toString().toLowerCase();
    for (String pattern : patternsToSkip) {
        line = line.replaceAll(pattern, "");
    }
    /* Format of the input data is
    * "Conference Acronym" \t "Year" \t "Conference Name" \t "Location" */
    /* Split by tab */
    StringTokenizer itr = new StringTokenizer(line, "\t");
    /* We need to get the conference's acronym in the first column*/
    if (itr.hasMoreTokens()) {
        confAcronym.set(itr.nextToken());
    } else {
        System.err.println("Bad line: " + line + "\n");
    }
    /* Skip the next two columns */
    for (int i = 0; i < 2; i++) {
        if (itr.hasMoreTokens()) {
            itr.nextToken();
        } else {
            System.err.println("Bad line: " + line + "\n");
            break;
        }
    }
    /* We need to process the "City" in the fourth column */
    if (itr.hasMoreTokens()) {
        city.set(itr.nextToken());
        /* Use "city" as key, and "conference acronym" as value */
        context.write(city, confAcronym);
        Counter counter = context.getCounter(CountersEnum.class.getName(),
            CountersEnum.INPUT_WORDS.toString());
        counter.increment(1);
    } else {
        System.err.println("Bad line: " + line + "\n");
    }
}
}
```

The reduce function of the Reducer looks like

```
/* We will use "Text" for the value field of the Reducer */
```

```

public static class TextSumReducer
    extends Reducer<Text,Text,Text,Text> {
    private Text result = new Text();

    public void reduce(Text key, Iterable<Text> values,
        Context context
        ) throws IOException, InterruptedException {
        String sum = "";
        for (Text val : values) {
            String conf = val.toString();

            /* Remove spaces at the end and beginning of the string */
            conf = conf.trim();
            /* Add the conference to the list if it is not yet present in the list */
            if (sum.indexOf(conf) == -1) {
                sum += conf + ", ";
            }
        }
        /* Remove the ", " at the end */
        sum = sum.trim();
        while ((sum.length() > 0) && (sum.charAt(sum.length()-1) == ',')) {
            sum = sum.substring(0,sum.length()-1);
        }
        result.set(new Text(sum));
        context.write(key, result);
    }
}

```

- For each conference regardless of the year (e.g., KDD), output the list of cities I used "conference acronym" (Text) as key and "city" (Text) as value. The output file has two columns: "Conference acronym" and "City list", which contains list of cities separated by ",".

The map function is as follows:

```

public void map(Object key, Text value, Context context
    ) throws IOException, InterruptedException {
    String line = (caseSensitive) ?
        value.toString() : value.toString().toLowerCase();
    for (String pattern : patternsToSkip) {
        line = line.replaceAll(pattern, "");
    }
    /* Format of the input data is
    * "Conference Acronym" \t "Year" \t "Conference Name" \t "Location" */
    /* Split by tab */
    StringTokenizer itr = new StringTokenizer(line,"\t");
    /* We need to get the conference's acronym in the first column */
    if (itr.hasMoreTokens()) {

```

```

        confAcronym.set(itr.nextToken());
    } else {
        System.err.println("Bad line: " + line + "\n");
    }
    /* Skip the next two columns */
    for (int i = 0; i < 2; i++) {
        if (itr.hasMoreTokens()) {
            itr.nextToken();
        } else {
            System.err.println("Bad line: " + line + "\n");
            break;
        }
    }
    /* We need to process the "City" in the fourth column */
    if (itr.hasMoreTokens()) {
        city.set(itr.nextToken());
        /* Use "conference acronym" as key and "city" as value */
        context.write(confAcronym, city);
        Counter counter = context.getCounter(CountersEnum.class.getName(),
            CountersEnum.INPUT_WORDS.toString());
        counter.increment(1);
    } else {
        System.err.println("Bad line: " + line + "\n");
    }
}
}
}

```

The reduce function is as following:

```

/* We will use "Text" for the value field of the Reducer */
public static class TextSumReducer
    extends Reducer<Text,Text,Text,Text> {
    private Text result = new Text();

    public void reduce(Text key, Iterable<Text> values,
        Context context
    ) throws IOException, InterruptedException {
        String sum = "";
        for (Text val : values) {
            String conf = val.toString();

            conf = conf.trim();
            /* Add the current city to the list if it is not yet present in the list */
            if (sum.indexOf(conf) == -1) {
                sum += conf + ", ";
            }
        }
    }
}

```

```

        /* Remove the ", " at the end */
        sum = sum.trim();
        while ((sum.length() > 0) && (sum.charAt(sum.length()-1) == ',')) {
            sum = sum.substring(0,sum.length()-1);
        }
        result.set(new Text(sum));
        context.write(key, result);
    }
}

```

4. For each city compute and plot a time series of #conferences per year
 I use “City + Year” (Text) as key and it occurrences (IntWritable) as value. The output file has three columns: “City”, “Year”, and “Number of conferences”.
 The map function is as follows

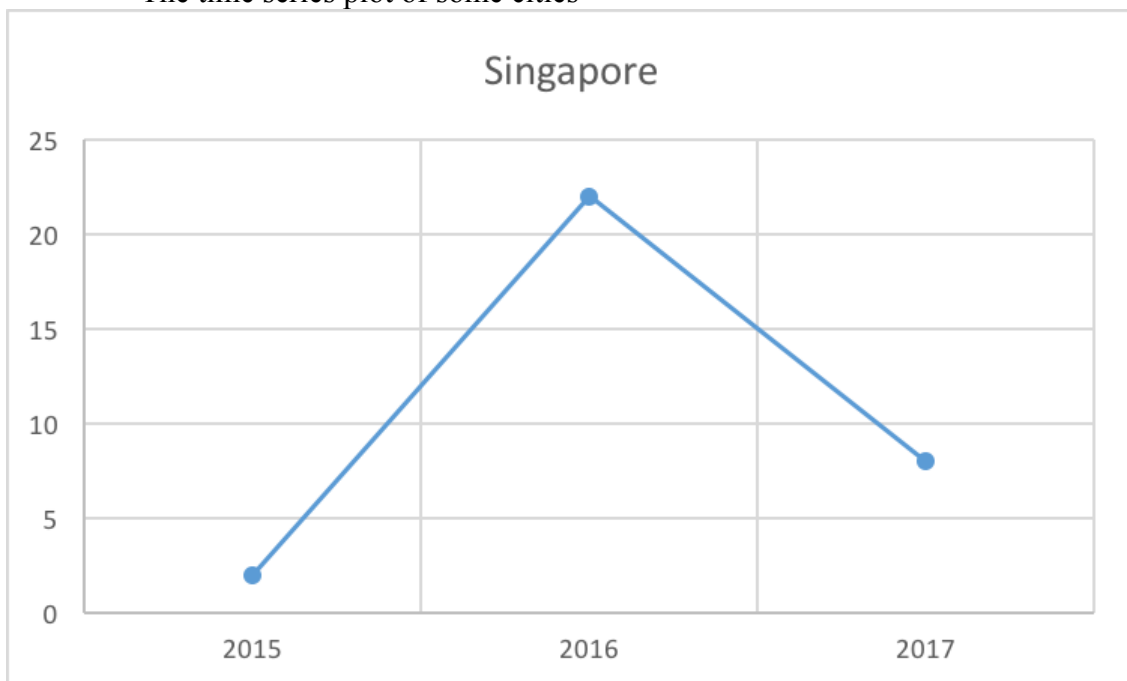
```

public void map(Object key, Text value, Context context
                ) throws IOException, InterruptedException {
    String cityStr = "";
    String yearStr = "";
    String line = (caseSensitive) ?
        value.toString() : value.toString().toLowerCase();
    for (String pattern : patternsToSkip) {
        line = line.replaceAll(pattern, "");
    }
    /* Format of the input data is
    * "Conference Acronym" "Year" "Conference Name" "Location" */
    /* Split by tab */
    StringTokenizer itr = new StringTokenizer(line, "\t");
    /* Skip the first column */
    if (itr.hasMoreTokens()) {
        itr.nextToken();
    } else {
        System.err.println("Bad line: " + line + "\n");
    }
    /* Get the year */
    if (itr.hasMoreTokens()) {
        yearStr = itr.nextToken().toString();
    } else {
        System.err.println("Bad line: " + line + "\n");
    }
    /* Skip the third column */
    if (itr.hasMoreTokens()) {
        itr.nextToken();
    } else {
        System.err.println("Bad line: " + line + "\n");
    }
    /* We need to process the "City" column */
}

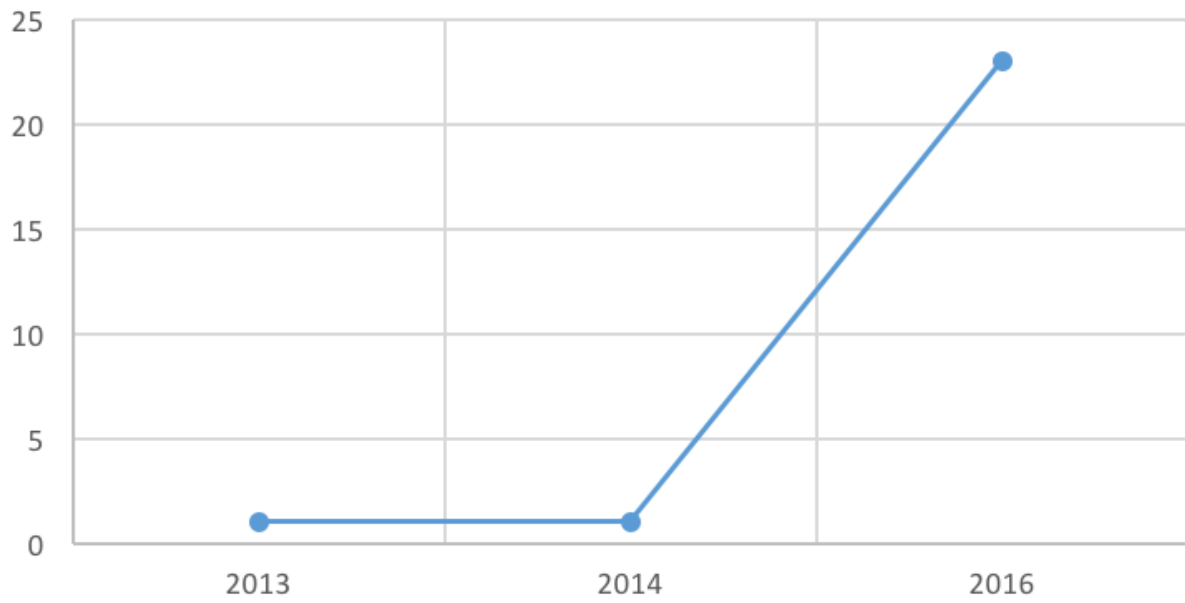
```

```
if (itr.hasMoreTokens()) {  
    cityStr = itr.nextToken().toString();  
    /* Use "city + year" as key and number of occurrences as value */  
    context.write(new Text(cityStr + "\t" + yearStr), one);  
    Counter counter = context.getCounter(CountersEnum.class.getName(),  
        CountersEnum.INPUT_WORDS.toString());  
    counter.increment(1);  
}  
}  
}
```

The time series plot of some cities



Barcelona



Montreal

