CS 2413 – Data Structures – Spring 2022 – Project 1

Due 11:59 PM, February 17, 2022

Dr Katia Papakonstantinopoulou

DESCRIPTION OF THE DATA STRUCTURE

In this project we deal with handling sparse matrices. We are interested in *nxn* matrices that represent the 'retweets' that take place among a set of *n* Twitter users. In particular, the element *A[i,j]* of matrix *A* represents the number of times user *i* has retweeted user *j* in a specific period. We store these matrices in a data structure called "Compact Retweet Matrix" (CRM), which is explained in detail below. In this project you are asked to define this data structure in a class, together with a set of methods that operate on it.

As an example, consider a set of 10 Twitter users and the following 10x10 matrix (consisting of 10 rows and 10 columns, each one numbered using integers from 0 to 9). The non-zero values are in red font just for the sake of clarity.

	0	1	2	3	4	5	6	7	8	9
0	0	0	12	0	5	0	0	0	0	0
1	0	0	0	0	0	0	4	0	0	0
2	0	7	0	0	0	0	0	0	6	0
3	0	0	2	0	0	0	13	1	0	0
4	0	0	0	0	0	0	0	0	18	0
5	0	15	0	0	0	0	0	54	0	17
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	38	0	85	0	0	92	0
8	0	0	0	0	0	0	0	0	0	0
9	37	29	0	0	0	0	0	0	20	0

We consider the CRM as a triplet of arrays (*values*, *rowPos*, and *colPos*) which collectively include all the above information.

In particular:

- The *values* array has size equal to the number of non-zero values in the matrix, and contains all these values, by the order they appear in the matrix. In the above matrix, there are 18 non-zero values. (The highlighting is used for the connection with the next array, please ignore it for now.)

values array:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
12	5	4	7	6	2	13	1	18	15	54	17	38	85	92	37	29	20

- The *rowPos* array has size equal to the number of users (10 in our case), and for each user it contains the position in values array in which the first nonzero value corresponding to this user appears. The highlighting colors are used in order to help you understand the relation between arrays *rowPos* and *values*.

rowPos array:

0	1	2	3	4	5	6	7	8	9
0	2	3	5	8	9	-1	12	-1	15

Therefore, rowPos[0] = 0 indicates that the non-zero values in row 0 of the matrix start being stored at position 0 of the *values* array. Indeed, it is values[0]=12 and 12 is the first non-zero element in row (user) 0 of the matrix. Similarly, rowPos[1] = 2 indicates that the non-zero values in row 1 of the matrix start at index position 2 of the *values* array. Indeed, at index position 2 (i.e., values[2]), the value is 4 which is the first non-zero value in row 1. Therefore, the non-zero values of user 0 start from position 0 of the *values* array and extend up to position 1, since in position 2 we start storing the non-zero values of user 1, and hence they are 12 and 5.
In the same spirit, rowPos[7] contains the value 12. This indicates that the first non-zero value in row 7 of the matrix can be found at values[12], which is 38. The non-zero values of user 9 extend from position 15 of the *values* array until, obviously, the end of it.
We use –1 in the *rowPos* array to indicate the fact that a row does not contain any non-zero values.

- The *colPos* array has size equal to the number of non-zero values in the matrix (18 in our case), and for each such value it contains the column of the matrix in which it appears.

colPos array:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
2	4	6	1	8	2	6	7	8	1	7	9	3	5	8	0	1	8

Your program should be reading a matrix and it should be storing it in a CRM data structure. Then it should be able to perform queries on this structure.

INSTRUCTIONS AND GUIDELINES

Input format:
For the purposes of this project, when we give a matrix as an input, we will be providing that in the sparse format, i.e., a vector (*row*, *col*, *value*) for each non-zero *value*, indicating the *row* and *col*umn of the matrix in which this *value* appears. In the first line of the input, we give the total

number of rows and the total number of columns of the matrix. For example, for the matrix above the input should be the following:

```
10 10         <- Total number of rows and columns
18            <- Total number of non-zero values
0 2 12        <- row number, column number and value. (We continue providing one such
0 4 5            vector for each non-zero value.)
1 6 4
2 1 7
2 8 6
3 2 2
3 6 13
3 7 1
4 8 18
5 1 15
5 7 54
5 9 17
7 3 38
7 5 85
7 8 92
9 0 37
9 1 29
9 8 20
```

Programming Objectives:

1. All code must be written in **standard C++** (so that it can be compiled by a g++ compiler).
2. **You have to submit**: A file named **HW1_CS2413.cpp** that contains all the code of this project.
3. **You are provided**: A file named template1.cpp including the template your submission should follow, as well as sample input and output files of your program. These files will be available in Module "Projects" in Canvas.
4. You will create a class named CRM (using the given template file) that has fields for the three arrays.
5. The class will have several methods, including *finding the most influential user* (i.e., the user whose posts have been retweeted the most times), *finding the most active user* (i.e., the user who has made the largest number of retweets), *ranking* the users in a descending order based on *how influential* they are, as well as *ranking* them based on *how active* they are. The latter two methods should compute a vector of all users, sorted according to their ranking.
6. All input will be read via **redirected input**. That is, you should not open a file inside the program.

7. Your program should be given as input a **matrix** in the format that was illustrated in the example above. Please refer to the sample input and output files given, and make sure to follow these formats exactly.
8. The **CRM class structure** should be as shown in the provided template file (you are responsible for fixing any syntax errors!). We have provided code for a few methods, and you need to make sure that they work correctly. You will also write code for other methods needed.
9. The structure of your **main** program is also provided, and you should use that in your project.

Redirected Input:
Please check out the **Visual Studio Installation and Setup guidelines for C++.doc** provided to you on canvas. Section 3 in the document has instructions on setting up the redirected input to Visual Studio.

Constraints:
1. Please save your submission file as "HW1_CS2413.cpp" ONLY
2. In this project, the only header you will use is #include <iostream> and using namespace std.
3. None of the projects is a group project. Consulting with other members of this class or seeking coding solutions from other sources including the web on programming projects is strictly not allowed and plagiarism charges will be imposed on students who do not follow this.

This file will be being updated, if necessary, to reflect any further clarifications that may be given to the students by the instructor or the TAs.