UNIVERSIDAD POLITÉCNICA DE YUCATÁN

Sparse Matrix

Project 1

Objective

The student will build linear structures for data storage and retrieval.

Instructions

Motivation

A network is composed of system's components often called nodes or vertices and the direct interactions between them, called links or edges. Figure 1 shows a small subset of (a) the internet, where routers are connected to each other; (b) Hollywood actor network, where two actors are connected if they played in the same movie; (c) protein-protein interaction network, where two proteins are connected if there is experimental evidence that they can bind to each other in the cell. All different systems have the same network representation using a graph. The number of vertices in the network is common denoted by N and the number of links by L.

A description of a network requires to track its link. One common representation is by using its adjacency matrix. The adjacency matrix of the network in Fig. 1 has N rows and N columns, its elements being (Fig. 2):

- $A_{ij} = 1$ if vertices i and j are connected to each other.
- $A_{ij} = 0$ if vertices i and j are not connected to each other.

Remember that A_{ij} represent the cell of the adjacency matrix in row i and column j. In Fig. 2, we can observe that there is a link between vertices 1 and 2 because $A_{1,2} = 1$.

In real networks L is much smaller than the maximum number of links in a network. In a network of N vertices, the maximum number of links is given by:

$$L_{max} = \frac{N(N-1)}{2}$$

In real networks only a tiny fraction of links in the matrix are nonzero. A matrix which contains very few non-zero elements are called **sparse matrix**.

In this project, you are going to build a computational representation of sparse matrix by using linked list and functions to perform matrix operations.

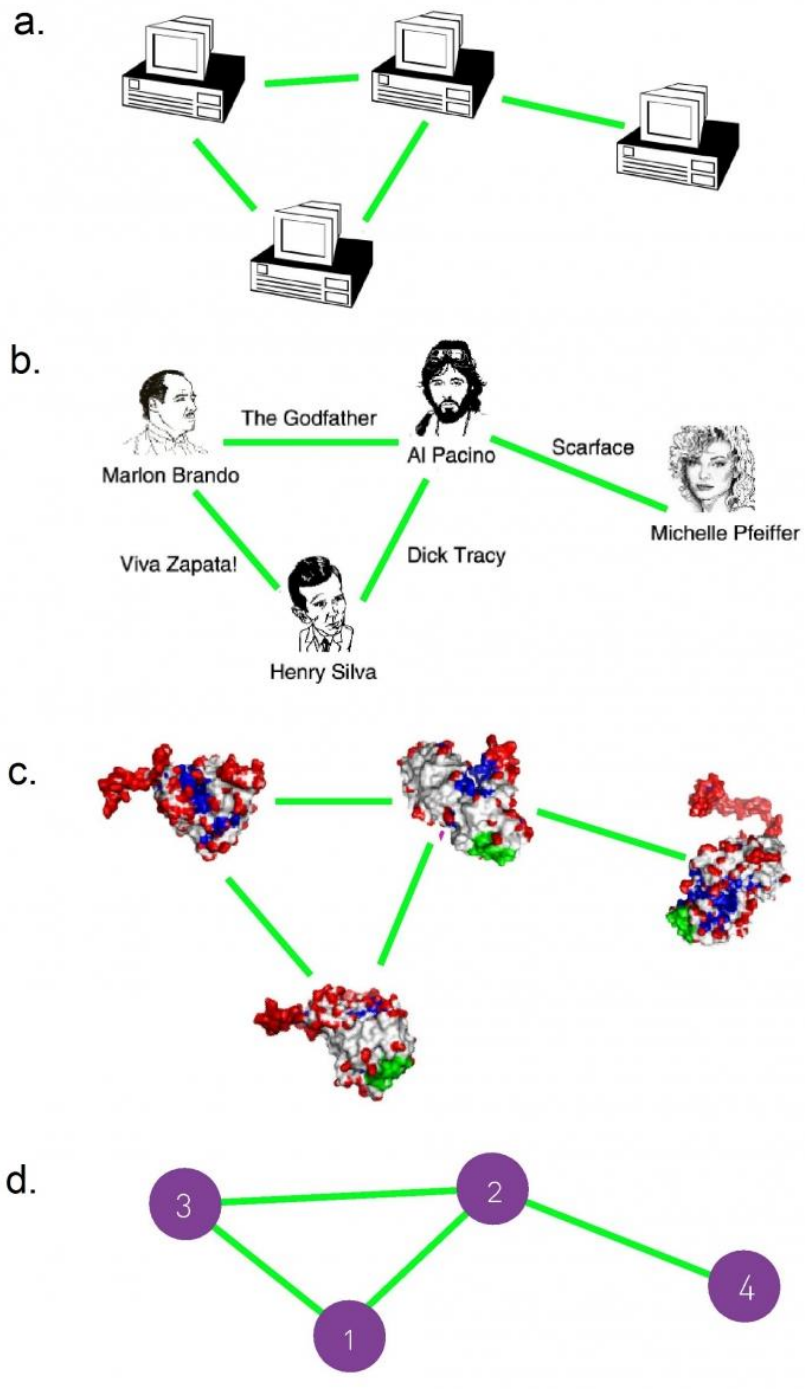
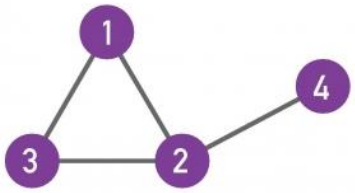

a.
b.
The Godfather
Marlon Brando
Al Pacino
Scarface
Michelle Pfeiffer
Viva Zapata!
Dick Tracy
Henry Silva
c.
d.
3
2
4
1

Fig. 1

a. **Adjacency matrix**

$$A_{ij} = \begin{matrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{matrix}$$

b. **Undirected network**



$$A_{ij} = \begin{matrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{matrix}$$

Fig. 2

Project Description

Part 1 – Data Structure Implementation

The program should use multi-linked circular linked list to represent sparse matrix. A linked list node is going to store the information of a vertex in the sparse matrix. The nodes should contain the following information (Fig. 3):

- `info`, the information contained in cell of the matrix.
- `row`, the row where info is stored.
- `column`, the column where column is stored.
- `nextRow`, is a pointer to the next node in `row` with relevant information (non-zero).
- `nextColumn`, is a pointer to the next node in `column` with relevant information (non-zero).

row	column	Info
nextRow	nextColumn	

Fig. 3

For example, consider the following matrix:

$$\begin{matrix} 3 & 0 & 2 \\ -1 & 0 & 0 \\ 0 & 0 & 5 \end{matrix}$$

Using the circular linked list, the previous matrix can be represented as Fig. 4. Observe that an extra row and column are included. All element in the extra row has zero in `row` and in the extra column has zero in `column`. Note that all the cells with zero are not included in the linked list representation of sparse matrix.

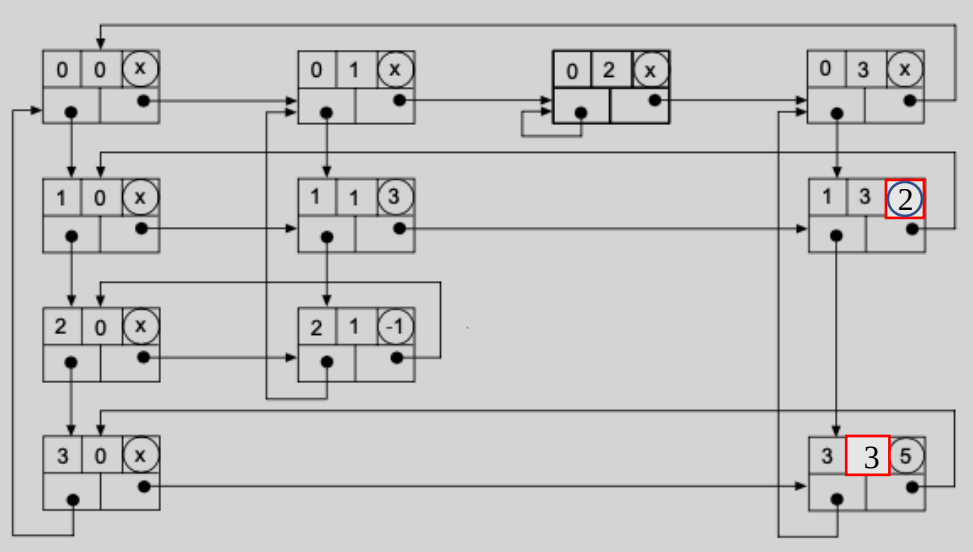


Fig. 4

Complete the following functions:

1. `createMatrix`, function to create a matrix with the zero row and column (extra row and column) given the number of rows and columns of the matrix to be stored.
2. `insert`, function to insert a node given its `row` and `column`. Display a warning in case that the `row` or `column` is out of range. In Fig. 4, after the creation of a new 3×3 matrix, four nodes are inserted: 3 at row 1 and column 1, 2, at row 1 and column 3, -1 at row 2 and column 1; and 5 at row 3 and column 3.
3. `delete`, function to delete a node given its `row` and `column`. It must return the deleted value. Display a warning in case that the `row` or `column` is out of range.

4 `assign`, function to assign a new value at a given row and column. ~~In case that the node does not exist, it must return `zero`.~~

5 `read`, function to read the value of a cell at a given row and colum. In case that the node does not exist, it must return `zero`.

6 `sum`, function to perform the addition of two matrices. In case that the addition is not viable, display a warning.

7 `product`, function to perform the multiplication of two matrices. In case that the multiplication is not viable, display a warning.

Part 2 - Shortest Path

Observe Fig. 2, the adjacency matrix indicates if two vertices are connected by a direct path. For example, $A_{1,2} = 1$ indicates that there is a direct path from vertex 1 to 2; $A_{4,2} = 1$ indicates that there is a direct path from vertex 4 to 2.

Powers of the adjacency matrix can be useful to understand the connectivity on the network. Consider the square of the adjacency matrix given in Fig. 2:

$$A^2 = A \times A = \begin{matrix} 2 & 1 & 1 & 1 \\ 1 & 3 & 1 & 0 \\ 1 & 1 & 2 & 1 \\ 1 & 0 & 1 & 1 \end{matrix}$$

Observe that $A^2_{1,4}$ = 1. It indicates that there is a path of length 2 from vertex 1 to 4. As you can see in Fig. 2 the path is $1 \rightarrow 2 \rightarrow 4$. On the other hand, $A^2_{1,1}$ = 2 means there are 2 possible paths from 1 to 1 with length of 2. Those paths are $1 \rightarrow 2 \rightarrow 1$ and $1 \rightarrow 3 \rightarrow 1$.

Now, consider the graph in Fig. 5, its adjacency matrix is given by:

$$A = \begin{matrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{matrix}$$

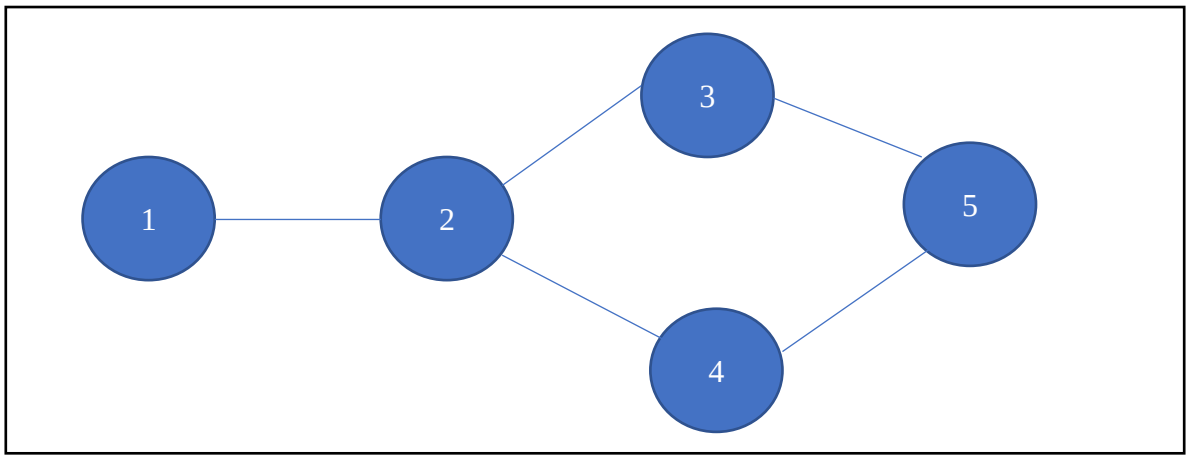


Fig. 5

Observe that there is no direct path from vertex 1 to 5 because $A_{1,5} = 0$. Now, let us check A^2:

$$A^2 = \begin{matrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 3 & 0 & 0 & 2 \\ 1 & 0 & 2 & 2 & 0 \\ 1 & 0 & 2 & 2 & 0 \\ 0 & 2 & 0 & 0 & 2 \end{matrix}$$

Observe that there is no path with length 2 from vertex 1 to 5 because $A^2_{1,5} = 0$. Now, let us check A^3:

$$A^3 = \begin{matrix} 0 & 3 & 0 & 0 & 2 \\ 3 & 0 & 5 & 5 & 0 \\ 0 & 5 & 0 & 0 & 4 \\ 0 & 5 & 0 & 0 & 4 \\ 2 & 0 & 4 & 4 & 0 \end{matrix}$$

Observe that there are two paths with length 3 from vertex 1 to 5 because $A^3_{1,5} = 2$. In Fig. 5, we can note that those paths are: $1 \rightarrow 2 \rightarrow 3 \rightarrow 5$ and $1 \rightarrow 2 \rightarrow 4 \rightarrow 5$. In summary, the shortest path which connect vertices 1 and 5 has length of 3.

Use your functions implemented in Part 1 to solve the following:

8 Store the following sparse matrix by using the implementation completed on Part 1:

$$\begin{matrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{matrix}$$

9 Create a function that given two vertices computes the length of the shortest path between them. For example, for the network in Fig. 5, given vertices 1 and 5, it must return 3.
10 Use your function in 7 to find the distance of the shortest path between the vertices:
   a. 1 and 2.
   b. 1 and 6.
   c. 1 and 8.
11 Make a video explaining your implementation and results of Part 1 and Part 2. All members must present the corresponding part of the project done per each one.

Submission

Submit the following before October 11th:

- Source Code of Part 1 and Part 2.
- Link to you video.

Teams

The maximum number of members per team is 4.

Grading Scheme

Part 1	50%
Part 2	30%
Video	20%

Suggested Reading

- Here is an interesting post that you can read about the graph concepts used in this project: https://www.sharetechnote.com/html/EngMath_Matrix_AdjacencyMatrix.html

- Chapter 2 Graph Theory of Network Science by A. Barabasi has a very nice introduction to the concepts used in this project. You can read it in the following link: http://networksciencebook.com/chapter/2#paths