ISEP
Algorithmics and Advanced programming
March 15th 2017

Tutorial course 3 : Graphs, an introduction

Instructions : Prepare a report including the source code and the results in a pdf file and upload it to moodle into the folder "Report Tutorial course"

1 Objectives

— Get familiar with graph data structures.
— Get familiar with structural properties of graphs.

2 Manipulating graph data

This tutorial will discuss implementing a graph data structure in Java.

As discussed in the lecture, there are two types of graph representation :
— Adjacency list representation.
— Adjacency matrix representation.

2.1 Adjacency list representation of an undirected graph

Define a class Graph containing three attributes : the order of the graph (number of nodes), the size of the graph (the number of edges) and an adjacency list representation of the graph.

```java
public class Graph {
    private final int N;
    private int M;
    private List<Integer>[] adj;
}
```

It is also possible (an even sometimes very useful necessary) to create a class *node* and a class *edge*. Then each entry of the adjacency list becomes a list of objects of class *node* (instead of a list of integer numbers).

The file *graph.txt* contains a small graph of an undirected and weighted graph. Open the file in a text editor and see how the file format looks like.

Each line in the file represents and edge. For example, the second line indicates that there is an edge between vertices 1 and 2.

Add the following functions to the class graph :

1. Initialize an empty graph of order N (input parameter) and 0 edges.
2. Initializes a graph from a specified input stream. You will test this function by reading the graph from the *graph.txt* file.
3. return the total order and the size of the graph.
4. a function called addEdge(int u, int v) that takes as input parameters two integers representing two vertex labels, the endpoints of the new edge. This function will add an edge between two existing nodes to the graph.

5. create a function Neighbors(int v) that takes a as input a given vertex and prints all the neighbors of that vertex.
6. Add a function that prints the Adjcency list representation of the graph.

Test all these functions with the graph contained the text file.

Now, you are going to study some structural properties of the graph. Create a function called Degree(int v) that returns for a given vertex v, its degree. Answer the following questions :

1. What is the average, minimal and maximal degree of the graph ? What is the edge-density ? Do you consider this graph dense or sparse ?
2. Are there any isolated nodes ? If yes, which ones ?
3. Are there any loops ?
4. Verify your answers by using the Neighbors(int v) function.

Finally, write a function that allows to read data graph from the keyboard input. The program will ask the user the total number of vertices, the total number of edges and user will have to type each edge as in the following example :

```
Enter the number of vertices:
5
Enter the number of edges:
4
Enter the edges in the graph : <to> <from>
1 2
```

2.2 Adjacency matrix representation of an undirected graph

Define a class GraphAdjMatrix containing three attributes : the order of the graph (number of nodes), the size of the graph (the number of edges) and an adjacency matrix.

```java
public class Graph {
    private final int N;
    private int M;
    private boolean[][] adj;
}
```

Create the following functions :

1. Initialize an empty graph of order N (input parameter) and 0 edges.
2. Initializes a graph from a specified input stream. All the entries of the matrix adj must be initialiazed. You will test this function by reading the graph from the *graph.txt* file.

Why is it preferably to use an adjacency list representation in practical contexts ?

3 Practical application

Now let us consider a bigger graph. A graph of a social network between 34 members of a karate club at a US university in the 1970s is the *Zachary Karate club network*. The club hired a karate instructor, Mr. Hi (node 1). There was a conflict between Mr. Hi and the club president, John A. (node 34) over the price of the lessons. The entire club became divided over this issue during the decision meetings before spliting completely into two parts. Mr. Hi was fired and created another club. About half of the active members joined him whereas the other half supported John. Not all individuals in the network were solidly members of one faction or the other. Some vacillated between the two ideological positions, and others were simply satisfied not to take sides. The *Zachary Karate club network* is given in the file *karate.txt*. Import the graph data and analyse the structure of this network. Why is it possible to say that the nodes 1 and 34 occupy a central position ?