ESTRUCTURA DE DATOS 2
Código ST0247

Laboratory practice No. 1: Graphs implementation

Felipe Ríos López
Universidad Eafit
Medellín, Colombia
friosl@eafit.edu.co

Santiago Gil Zapata
Universidad Eafit
Medellín, Colombia
sgilz@eafit.edu.co

3) Practice for final project defense presentation

3.1. We start by reading the txt which contains all the data, and then, we ignore the data that is irrelevant by reading every line with arrays, taking only the first string that contains the ID (nodes) and then we get the edjes. After that, we take those nodes and destinations from the second part of the file , arcs in a dictionary that contains the weight in order to get a listo f destinys and distance from a certain point.
3.2. When there is a matrix so big it is very probable that we would get an error of OutOfMemory and we are taking into acccount that those Will be at least 2 loop, 300.000^2 = 90'000'000.000
3.3. Taking the ID like it's the first key; as mentioned before, with a dictionary.

3.4. We used a dictionary that inside of them contained an array with every of the positions of the edges where they connected, and the logic of the program is that first we want to obtain the subjacents of every edge and compare them to the other rest of the edges, meaning that if an edge connected to an edge and also to other edge that also connected to the first edge mentioned, it would not be posible to be bicolored.

3.5 T(n) = (C1 + C2) + m + n^3 = O(n^3) in the worst cases.

3.6 n is the number of data, m is the number of this data that will be proccessed

4) Practice for midterms

4.1 a

	0	1	2	3	4	5	6	7
0				1	1			
1	1		1			1		
2					1		1	
3								1
4			1					
5								
6			1					
7								

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

4.2
0-> [3,4]
1->[0,2,5]
2->[4,6]
3->[7]
4->[2]
5->[]
6->[2]
7->[]

5) Recommended reading (optional)

Graphs (resume)

Introduction:

Graphs are data structures which have forms dictated by a physical problem. For instance, nodes in a graph may represent cities, while edges may represent airline flight routes, between those cities.

Graph parts:

A graph, essentially, has two kinds of elements:

- Vertex (nodes):
  In most situations, a vertex represents some real-world object, and the object must be described using data items.
- Edges:
  An edge is the relationship between a vertex with one or more other vertices. Sometimes, an edge can has the weight of this relationship.

Adjacency:

A vertex is said to be *adjacent* to another if they are connected by a single edge.

Paths:

A path is a sequence of edges that connect two vertices, passing through zero or more other vertices. There can be more than one path between two vertexes.

Connected graphs:

A graph is said to be *connected* if there is at least one path from every vertex to very other vertex.
A *non-connected* graph has several *connected components* (connection vertex to vertex), but there is not any connection between these components.

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

Directed and weighted graphs:

A graph is *directed* when its edges have a direction. It means that you can go in only one direction along an edge; from A to B but not from B to A, as on a one-way street. A directed edge is normally shown as an arrow.
In some graphs, edges are given with a weight, number that can represent the physical distance between two vertices, or the time it takes to get from one to another, or how much it costs (The weight can represent different types of values depending on the purpose of our graph).

Implementation of graphs

Adjacency Matrix:

It is a two-dimensional array which contains the elements indicate whether an edge is connecting two vertices. For a graph that has N vertices, the adjacent matrix is an NxN array.
The vertices are the head for both rows and columns. The edge between two vertices is indicated by 1, and its absence is indicated by 0. If the graph is weighted, existence of an edge is indicated by its weight. For instance, if two vertices, A and B are connected by an edge with W (weight) equal to 34, then 34 is the value stored in the *adjacency matrix*.

Adjacency list:

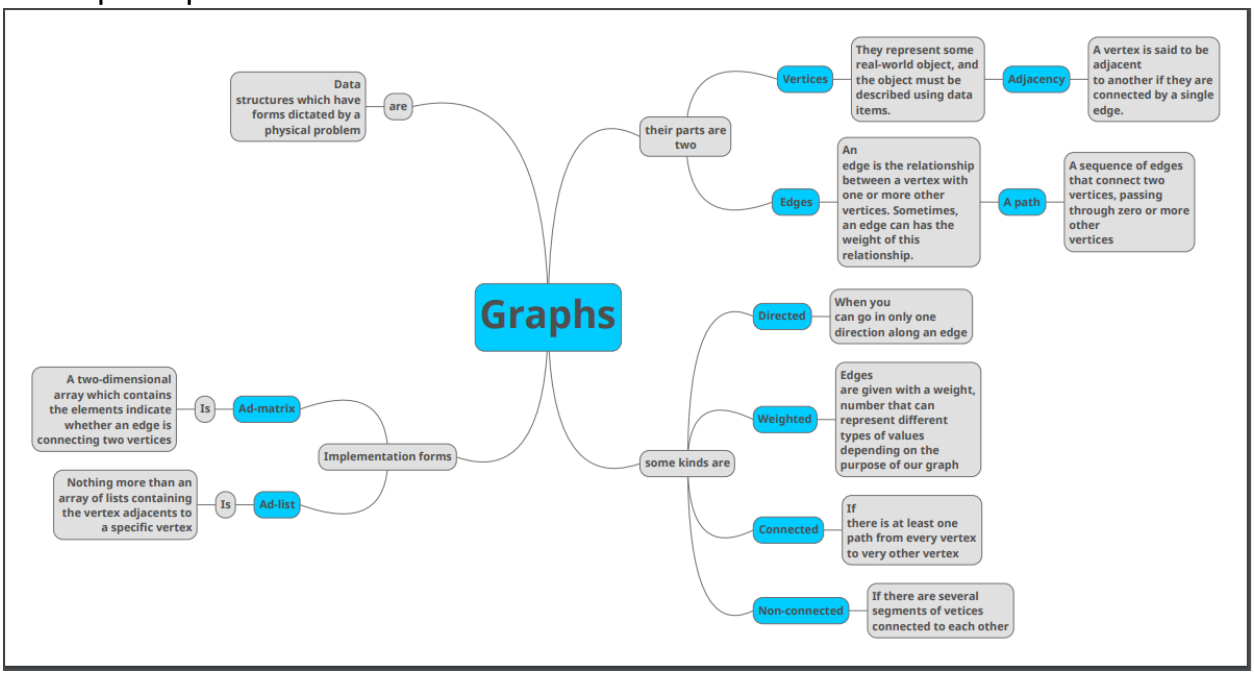
An *adjacency list* is the other way to represent a graph. It is nothing more than an array of lists. In every position of the array, there is a vertex as the head of a list, and the items in that list are the adjacent vertices of the head vertex. In general, when we are implementing a weighted graph by *adjacency list*, the items contained in every list are pairs DESTINATION VERTEX – WEIGHT.

Reference

Taken from:
Robert Lafore, *"Data Structures and Algorithms in Java (2nd edition)"*, Chapter 13: Graphs. 2002

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

Concept map:



6) Team work and gradual progress

6.1

Integrante	Fecha	Hecho	Haciendo	Por hacer
Felipe Ríos	2/02/2018	Punto 2	Comentar código	Informe
Santiago Gil	2/02/2018	Punto 1	Comentar código	Lectura y mapa
Felipe Ríos	3/02/2018	Revisión código	Punto 4	Informe y correccion
Santiago Gil	3/02/2018	Revisión código	Lectura y mapa	Revisar código de Felipe

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473