

**MATH 211**

**Graph Representation**

Abdelrahman Yasser Hussien 211000008

Youssef Hatem Mostafa 211001689

Deif Mohamed 211001739

Youssef El Naggar 211001713

**Dr. Mustafa Elsayed Abdulsalam**

Eng. Salma Khalid Ali

Eng. Mohamed Youssef El Shahed

Eng. Ahmad Ahmad Muhammad

12/28/2022

**Table of Contents**

**Abstract…………………………………………………………………………………………...**3

**1.Introduction…………………………………………………………………………………….**3

* 1. **Definition of Graphs…………………………………………………………………**3
  2. **Graphs Representation………………………………………………………………**3
  3. **Types of Graphs……………………………………………………………………**..4

**2.Methodology……………………………………………………………………………………**5

**2.1 Sequential representation of graph…………………………………………………**5

**2.1.1 Mathematical Definition…………………………………………………..**.5

**2.2 Linked representation of graph……………………………………………………..**5

**2.2.1 Mathematical Definition…………………………………………………...**5

**3.Results…………………………………………………………………………………………..**6

**3.1 Pros & Cons of Sequential Representation………………………………………...**6

**3.2 Pros & Cons of Linked Representation…………………………………………….**7

**4.Conclusion……………………………………………………………………………………...7**

**5.References………………………………………………………………………………………**8

**Abstract**

This study compares the two primary methods for storing graph representations in computer memory, namely adjacency matrix representation and adjacency list representation. The lesson starts off with defining graphs and going over the many kinds of graphs that are out there. The findings section assesses the benefits and drawbacks of each representation after the technique section defines adjacency matrix and adjacency list representation mathematically. The findings are summarized in the conclusion, which emphasizes the significance of considering the particular requirements and characteristics of the graph being represented when choosing which representation method to employ.

**Introduction**

* 1. **Definition of Graphs**

As a data structure with a limited number of nodes (also known as vertices and edges), a graph is described. As an ordered pair, they are shown as G(V, E), where V(u,v) and E (u,v). The pair in V reveals two vertices, while the pair in E reveals an edge connecting the graph nodes u and v. Moving from vertex u to vertex v may also cost anything for the edges. Many actual problems are represented symbolically in graphs, which aid in the visualization of the problem's resolution by helping us build the proper algorithm. It is understood that they are mathematical constructions. They are intended to show the pairings of various items. The movement of structures between entities is depicted on a graph.

* 1. **Graph Representations**

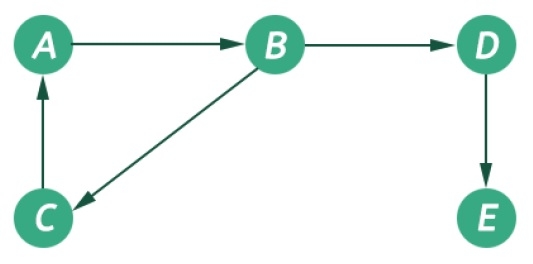
A graph is a type of data structure made up of groups of vertices (also known as nodes) and edges. Graphs can be stored in the computer's memory in two different ways:

* Adjacency Matrix Representation
* Adjacency List Representation
  1. **Types Of Graphs**

A graph consists of vertices, also known as nodes, and edges. In a graph, certain types of nodes or vertices exist, for example:

* Undirected: An undirected graph is one that contains all of its edges in a bi-directional arrangement, meaning that none of its edges are directed in any particular direction. Timeline

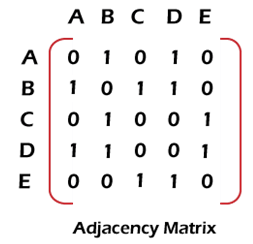
  Description automatically generated
* Directed: The edges linking the nodes in a directed graph, also known as a unidirectional graph, point in a single direction.



**Methodology**

* 1. ***Sequential representation of graph***

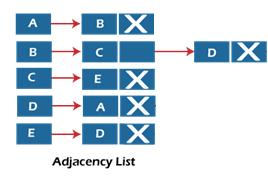
***2.1.1 Mathematical Definition***

****An adjacency matrix is a way to show which vertices of a network are close to which other vertices in mathematics and computer science. Specifically, each graph has a distinct adjacency matrix that differs from the adjacency matrices of all other graphs (up to permuting rows and columns). The adjacency matrix in the situation of a finite simple graph is a (0, 1)-matrix with zeros along its diagonal. As mentioned by (Singh & Sharma,2012)“Suppose G is an undirected graph. Then the adjacency matrix A of G will be a symmetric matrix, i.e., one in which aij = aji for every i and j. This follows from the fact that each undirected edge [u, v] corresponds to the two directed edges (u, v) and (v, u).”

***2.2 Linked representation of graph***

***2.2.1 Mathematical Definition***

An adjacency list in graph theory is a list representation of every edge or arc in a graph. If the graph is directed, each entry is a tuple of two nodes, one of which represents the source node and the other the destination node of the relevant arc. If the graph is undirected, each entry is a set (or multiset) of two nodes representing the two ends of the corresponding edge.



**Results**

**3.1 *Pros & Cons of Sequential Representation***

**Pros:** Working with an adjacency matrix is quite convenient. An edge may be added or removed in O(1) time, and it takes the same amount of time to determine whether there is an edge connecting two vertices.

**Cons:** 1-Adjacency matrix is optimal for dense graphs, but for sparse ones it is superfluous. Adjacency Matrix consumes huge amount of memory for big graphs like Fibonacci diagrams. 2-The adjacency matrix's next problem is that many algorithms require you to know the edges that are close to the current vertex. Moreover, the adjacency matrix demands a lot of work to add or remove a vertex. It is not essential if a graph is solely used for analysis, but if you want to build a completely dynamic structure, utilizing an adjacency matrix makes it quite slower for large graphs

**3.2 Pros & Cons of Linked Representation**

**Pros:** Although adjacent lists and adjacency matrices have different ways of storing graphs, the distinction gets smaller as a graph gets denser. The ability to obtain the list of neighboring vertices using an adjacent list in O(1) time is the next benefit, which is significant for some algorithms.

**Cons:** Adjacent lists prevent us from creating an effective solution if a dynamic change in the number of vertices is needed. While adding a new vertex may be accomplished in O(V), removing one requires O(E) complexity.

**Conclusion**

Considering the data given, adjacency matrices are better suited for dense graphs because they efficiently add and remove edges in O(1) time. However, for large, sparse graphs, they could use up a lot of RAM. However, since adding a new vertex has an O(V) complexity and removing one has an O(E) complexity, adjacency lists may not be as effective for dynamic changes in the number of vertices. Adjacency lists enable efficient retrieval of nearby vertices in O(1) time. Therefore, while choosing between an adjacency matrix and an adjacency list, it is crucial to consider the requirements and characteristics of the graph being represented.

**References**

Pedamkar, P. (2021, March 3). *Graph Representation*. EDUCBA. <https://www.educba.com/graph-representation/>

*Graph and its representations*. (n.d.). <https://www.tutorialspoint.com/graph-and-its-representations>

*Graph Representation - javatpoint*. (n.d.). www.javatpoint.com. <https://www.javatpoint.com/graph-representation>

Singh, H., & Sharma, R. (2012). Role of adjacency matrix & adjacency list in graph theory. *International Journal of Computers & Technology*, *3*(1), 179-183. [Role-of-Adjacency-Matrix-Adjacency-List-in-Graph-Theory.pdf (researchgate.net)](https://www.researchgate.net/profile/Harmanjit-Singh-2/publication/274362141_Role_of_Adjacency_Matrix_Adjacency_List_in_Graph_Theory/links/551be5540cf2fe6cbf75f9c7/Role-of-Adjacency-Matrix-Adjacency-List-in-Graph-Theory.pdf)