Data Structure and Algorithm

Laboratory Activity No. 10

Intro to Graphs

|  |  |
| --- | --- |
| *Submitted by:* | *Instructor:* |
| **MANONGSONG, KEN R.** | **Engr. Maria Rizette H. Sayo** |

10, 11, 2025

# Objectives

Introduction

A graph is a visual representation of a collection of things where some object pairs are linked together. Vertices are the points used to depict the interconnected items, while edges are the connections between them. In this course, we go into great detail on the many words and functions related to graphs.

An undirected graph, or simply a graph, is a set of points with lines connecting some of the points. The points are called nodes or vertices, and the lines are called edges.

A graph can be easily presented using the python dictionary data types. We represent the vertices as the keys of the dictionary and the connection between the vertices also called edges as the values in the dictionary.

A diagram of a triangle with green dots

AI-generated content may be incorrect.

Figure 1. Sample graph with vertices and edges

This laboratory activity aims to implement the principles and techniques in:

* To introduce the Non-linear data structure – Graphs
* To discuss the importance of Graphs in programming

# Methods

* 1. Discuss the following terms related to graphs:
     1. Undirected graph
     2. Directed graph
     3. Nodes
     4. Vertex
     5. Degree
     6. Indegree
     7. Outdegree
     8. Path
     9. Cycle
     10. Simple Cycle

# Results

1. **Undirected Graph**

An undirected graph is a collection of vertices or nodes and a set of edges where the edges have no direction. An edge connecting vertex A and vertex B signifies a mutual, two-way relationship, meaning you can traverse from A to B or from B to A. (Example: A social network where friendship is always mutual.)

2. **Directed Graph**

A directed graph is a collection of vertices and a set of edges where each edge has a specific direction, usually indicated by an arrow. An edge from vertex A to vertex B is an ordered pair, meaning the relationship is one-way: you can move from A to B, but not necessarily from B to A. (Example: A flow chart showing precedence, or a Twitter network where following is a one-way relationship.)

3. **Nodes**

Nodes are the fundamental entities or objects that make up a graph. They represent the data points or entities being connected.

4. **Vertex**

A vertex (plural: vertices) is an individual point in a graph. The set of all vertices is typically denoted as *V*.

5. **Degree**

The degree of a vertex , denoted , is the total number of edges connected to that vertex.

* In an undirected graph, it is simply the count of edges incident to .
* In a directed graph, the degree is the sum of the indegree and outdegree ().

6. **Indegree**

The indegree of a vertex , denoted , is used only in directed graphs. It is the count of edges that terminate at (come into) the vertex .

7. **Outdegree**

The outdegree of a vertex , denoted , is used only in directed graphs. It is the count of edges that originate at (go out from) the vertex .

8. **Path**

A path in a graph is a sequence of alternating vertices and edges that starts and ends with a vertex, such that each edge connects its preceding vertex to its succeeding vertex in the sequence.

9. **Cycle**

A cycle is a path that starts and ends at the same vertex, and has at least one edge. It is a closed walk where the first and last vertex are the same, and all edges in the sequence are distinct.

10. **Simple Cycle**

A simple cycle (or simple circuit) is a cycle in which the only repeated vertex is the starting and ending vertex. In other words, no other vertex is revisited during the traverse. (Example: In a graph with vertices A, B, C, the sequence A → B → C → A is a simple cycle. The sequence A → B → C → B → A is a cycle but not a simple cycle because vertex B is repeated (B is visited twice in the middle).

# Conclusion

The fundamental concepts of graph theory were properly discussed, establishing the graph as a way for data structure used to model relationships between entities. The terminologies were defined, highlighting the structural components like Nodes/Vertices and the nature of their connections via Edges. The distinction between Undirected Graphs and Directed Graphs is crucial for modeling different real-world scenarios. Lastly, Degree, Indegree, and Outdegree shows the connectivity of individual vertices, while Path, Cycle, and Simple Cycle define how entities can be traversed or related over a sequence of connections.

**References**

[1] Co Arthur O.. “University of Caloocan City Computer Engineering Department Honor Code,” UCC-CpE Departmental Policies, 2020.