GRAPH THEORY

AUTHORS:

-Anir Agountaf

-Muhammad Hassaan Shafique

PROFESSOR:

M. Soufian Benamor

COURSE:
Mathematics for Data Science

PROGRAM:
Master of Science in Data Management

Academic Year 2025-2026

Project Report

7 December 2025

Abstract

Graph theory is a foundational area of discrete mathematics concerned with the study of networks formed by nodes (vertices) and the connections (edges) between them. This report provides an in-depth overview of core graph-theoretic concepts, including graph representations, connectivity, traversals, shortest-path algorithms, tree and cycle structures and properties of directed and weighted graphs. Beyond these fundamentals, the report emphasizes the growing relevance of graph theory in modern computational fields, particularly artificial intelligence (AI).

In AI, graphs serve as powerful modeling tools for capturing complex relationships within data. Knowledge graphs represent structured information that supports reasoning and semantic understanding in intelligent systems. Graph algorithms enable efficient navigation and inference over these structures, contributing to advancements in natural language processing, recommendation systems and search technologies. Additionally, the emergence of graph-based machine learning methods—such as Graph Neural Networks (GNNs)—has expanded the role of graph theory in pattern recognition, social network analysis, molecular prediction and other areas where relational dependencies are critical.

By examining both classical theory and contemporary AI applications, this project highlights how graph theory provides the mathematical framework necessary for representing, analyzing and learning from interconnected data. The report concludes that as AI systems increasingly rely on relational information, the importance of graph theory will continue to grow, bridging the gap between mathematical abstraction and real-world intelligent technologies.

Keywords: Graph Theory, Data Science, Artificial Intelligence, Graph Neural Networks, Network Analysis, Machine Learning, Algorithms

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Graph theory is a branch of discrete mathematics concerned with the study of structures composed of **vertices** (nodes) and **edges** (connections). Although originating with Euler's 1736 solution to the Königsberg Bridge Problem [1], it has evolved into a major mathematical discipline with extensive applications in modern computer science, data science, network analysis and artificial intelligence.

1.1 Why Graph Theory Matters in Data Science

Many datasets today are relational rather than tabular. Examples include:

- Social networks — users connected via friendships.
- Web graphs — websites connected by hyperlinks.
- Transportation systems — cities connected by routes.
- Biological networks — proteins interacting with one another.

Traditional statistical methods often fail to capture this relational structure. Graph theory provides:

- Mathematical tools for modeling connected systems.
- Algorithms for efficient search, traversal and pattern discovery.
- Foundations for emerging fields such as Graph Neural Networks (GNNs) [2].

1.2 Basic Graph Concept Illustration

Figure 1.1 shows a basic undirected graph.

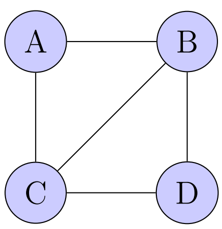

Figure 1.1: A sample undirected graph with four vertices.

1.3 Structure of This Report

This report is divided into five chapters:

- **Chapter 1**: Introduction to graph theory and its relevance.
- **Chapter 2**: Formal mathematical foundations and graph structures.
- **Chapter 3**: Classical graph algorithms used in data science.
- **Chapter 4**: Advanced topics including spectral graph theory and GNNs.
- **Chapter 5**: Conclusions and implications for data-driven applications.

Chapter 2

Fundamentals of Graph Theory

This chapter establishes the mathematical foundations needed to work with graph structures in data science. It introduces formal graph definitions, graph types, matrix representations and key structural concepts such as degree, connectivity and cycles.

2.1 Basic Definitions

Definition 2.1. A **graph** is an ordered pair $\mathcal{G} = (V, E)$ where:

$$V = \{v_1, \ldots, v_n\} \quad \text{and} \quad E \subseteq V \times V.$$

Edges may be **directed**, **undirected**, **weighted** or **unweighted**. Applications like social networks typically use undirected graphs, while recommendation systems or web link structures use directed, weighted graphs [3].

2.2 Degree and Neighborhood

For an undirected graph:

$$\deg(v) = |\{u \in V : (u, v) \in E\}|.$$

For a directed graph:

$$\deg^+(v) = \text{out-degree}, \qquad \deg^-(v) = \text{in-degree}.$$

2.3 Graph Representations

Graphs can be represented in multiple ways.

2.3.1 Adjacency Matrix

$$A_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

Table 2.1: Adjacency Matrix of a Simple Graph

	A	B	C	D
A	0	1	1	0
B	1	0	1	1
C	1	1	0	1
D	0	1	1	0

2.3.2 Adjacency List

For sparse graphs, adjacency lists are far more memory-efficient.

2.4 Graph Traversal Algorithms

Traversal algorithms are essential in data science — from exploring social networks to crawling web graphs.

Below are corrected versions of BFS and DFS in proper algorithm floats.

Algorithm 1 Breadth-First Search (BFS)

procedure BFS(G, s)
    mark s as visited
    enqueue s
    while queue not empty **do**
        $v \leftarrow$ dequeue()
        for each neighbor u of v **do**
            if u not visited **then**
                mark u
                enqueue u
            end if
        end for
    end while
end procedure=0

Algorithm 2 Depth-First Search (DFS)

```
procedure DFS(G, v)
    mark v as visited
    for each neighbor u of v do
        if u not visited then
            DFS(G, u)
        end if
    end for
end procedure=0
```

2.5 Connectivity and Components

A graph is **connected** if every pair of vertices is linked by some path. Connected components can be identified efficiently with BFS or DFS.

$$\text{Number of connected components} = k$$

2.6 Cycles and Trees

A **tree** is a connected, acyclic graph. Fundamental property:

$$\text{A tree with } n \text{ vertices has } n - 1 \text{ edges.}$$

Trees are widely used in machine learning (e.g., decision trees, hierarchical clustering).

Chapter 3

Network Science and Complex Networks

Network science extends classical graph theory to the study of large, real-world networks. Such networks often exhibit properties like clustering, heavy-tailed degree distributions, and small-world behavior [3].

3.1 Random Graph Models

3.1.1 Erdős–Rényi Model

In the $G(n, p)$ model, each pair of vertices is connected with probability p. This simple model facilitates theoretical analysis but rarely matches real data.

3.1.2 Watts–Strogatz Model

Designed to reproduce the "small-world" phenomenon: high clustering and short path lengths are typical of social systems.

3.1.3 Barabási–Albert Model

Models scale-free networks where degree distributions follow a power law:

$$P(k) \propto k^{-3}.$$

Such networks arise in the web graph, biological systems, and city transportation [?].

3.2 Centrality Measures

Centrality quantifies importance of nodes in a network.

3.2.1 Degree Centrality

Measures local influence:

$$C_D(v) = \deg(v).$$

3.2.2 Closeness Centrality

Measures how close a node is to others:

$$C_C(v) = \frac{1}{\sum_u d(v,u)}.$$

3.2.3 Betweenness Centrality

Measures how often a node lies on shortest paths:

$$C_B(v) = \sum_{s,t} \frac{\sigma_{st}(v)}{\sigma_{st}}.$$

3.2.4 Eigenvector Centrality and PageRank

Eigenvector centrality measures influence recursively: high-scoring nodes connect to high-scoring neighbors. PageRank modifies this with damping for web ranking [?].

3.3 Community Detection

Communities are densely connected node groups.

3.3.1 Modularity

Measures how well a partition captures community structure:

$$Q = \frac{1}{2m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{2m}) \delta(c_i, c_j).$$

Algorithms such as Louvain and Leiden optimize modularity.

3.4 Summary

This chapter introduced models and measures that help analyze real network behavior. Network science provides tools essential for identifying influential nodes, community structure, and overall graph organization.

Chapter 4

Advanced Graph Theory Concepts

This chapter covers modern graph concepts highly relevant to machine learning and data science.

4.1 Spectral Graph Theory

Spectral graph theory studies the eigenvalues and eigenvectors of matrices such as the Laplacian:

$$L = D - A.$$

The second-smallest eigenvalue λ_2, known as the **Fiedler value**, measures graph connectivity.

4.2 Random Walks on Graphs

Random walks are used in:

- PageRank [5]
- Node2Vec embeddings
- Semi-supervised learning on graphs

$$P_{ij} = \frac{A_{ij}}{\deg(i)}.$$

4.3 Graph Neural Networks (GNNs)

GNNs generalize convolution to irregular graph domains. A typical GCN layer [2] is:

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}\right).$$

Applications:

- Molecule property prediction
- Fraud detection
- Recommender systems

4.4 Knowledge Graphs

Knowledge graphs represent semantic relationships using triples:

$$(h, r, t)$$

where h is the head entity, t is the tail and r is the relation. They power modern AI systems including Google Knowledge Graph and large language models.

Chapter 5

Advanced Topics and Future Trends

As graph-based data becomes increasingly common, advanced topics have emerged to address scalability, expressiveness and dynamic behavior.

5.1 Graph Databases

Graph databases store data as interconnected entities, enabling fast traversal and pattern matching. Systems like Neo4j and TigerGraph support the Cypher and GSQL languages:

```
MATCH (u)-[:LIKES]->(i) RETURN i.
```

They support recommendation systems, identity resolution and enterprise knowledge management.

5.2 Hypergraphs

Hypergraphs generalize edges to connect multiple vertices:

$$e = \{v_1, v_2, \dots, v_k\}.$$

They model group interactions such as group chats, co-authorship networks and multi-protein complexes.

5.3 Temporal and Dynamic Graphs

Many real networks evolve over time: new users join, edges appear/disappear and weights change.

Dynamic graph models capture:

- evolving social relationships,
- time-varying traffic networks,
- real-time financial networks.

5.4 Graph Signal Processing (GSP)

GSP extends signal processing tools to graph-structured data.

Graph Laplacian:

$$L = D - A.$$

Graph Fourier Transform:

$$\hat{x} = U^T x,$$

where U contains Laplacian eigenvectors.

Used in sensor networks, recommendation systems and denoising.

5.5 Scalable Graph Computing

Handling billions of edges requires distributed frameworks like:

- Apache Spark GraphX,
- Google Pregel,
- DGL and PyTorch Geometric for GNNs.

These systems enable industrial-scale graph analytics.

5.6 Future Trends

Emerging research topics include:

- **Explainable GNNs** — interpreting learned patterns.
- **Neural-symbolic systems** combining logic and ML.
- **Real-time graph learning** for streaming data.
- **Foundational graph models** analogous to LLMs.

5.7 Summary

Advanced graph techniques support scalable, expressive and dynamic graph analytics essential to next-generation AI systems.

5.8 Conclusion

Graph theory offers a powerful mathematical framework for representing interconnected data, making it indispensable in modern data science. In this report, we explored fundamental graph concepts, classical algorithms, spectral methods and cutting-edge applications such as Graph Neural Networks and knowledge graphs. These methods enable us to capture relationships that cannot be modeled using traditional vector-based approaches. Classical algorithms like BFS, DFS, Dijkstra and MSTs continue to serve as essential building blocks for network analysis, optimization and search systems. At the same time, advanced topics including Laplacian eigenvectors, random walks and neural message passing highlight how graph theory is shaping the next generation of artificial intelligence tools.

As data becomes increasingly relational—whether through social networks, communication networks, molecular structures or web graphs—the significance of graph-based modeling will only continue to grow. GNNs and knowledge graphs already demonstrate how graph theory enables deeper reasoning, more effective predictions and richer representations. The field stands at the crossroads of mathematics, computer science and intelligent systems and its influence will only intensify in the coming years. Understanding graph theory is therefore not only academically valuable but also essential for building robust, scalable and intelligent data-driven technologies.

Bibliography

[1] L. Euler, “The seven bridges of königsberg,” 1736.

[2] T. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” *ICLR*, 2017.

[3] M. Newman, *Networks: An Introduction.* Oxford University Press, 2010.

[4] S. Fortunato, “Community detection in graphs,” *Physics Reports*, 2010.

[5] S. Brin and L. Page, “The pagerank citation ranking: Bringing order to the web,” 1998.