National University of Computer & Emerging Sciences

Karachi Campus

Project Report

MT3001 – Graph Theory

Made By:

Sufiyaan Usmani (21K-3195)

Muhammad Shahmir Raza (21K-3158)

Yousuf Ahmed (21K-4594)

Course Instructor:

Engineer Usama Antuley

Introduction:

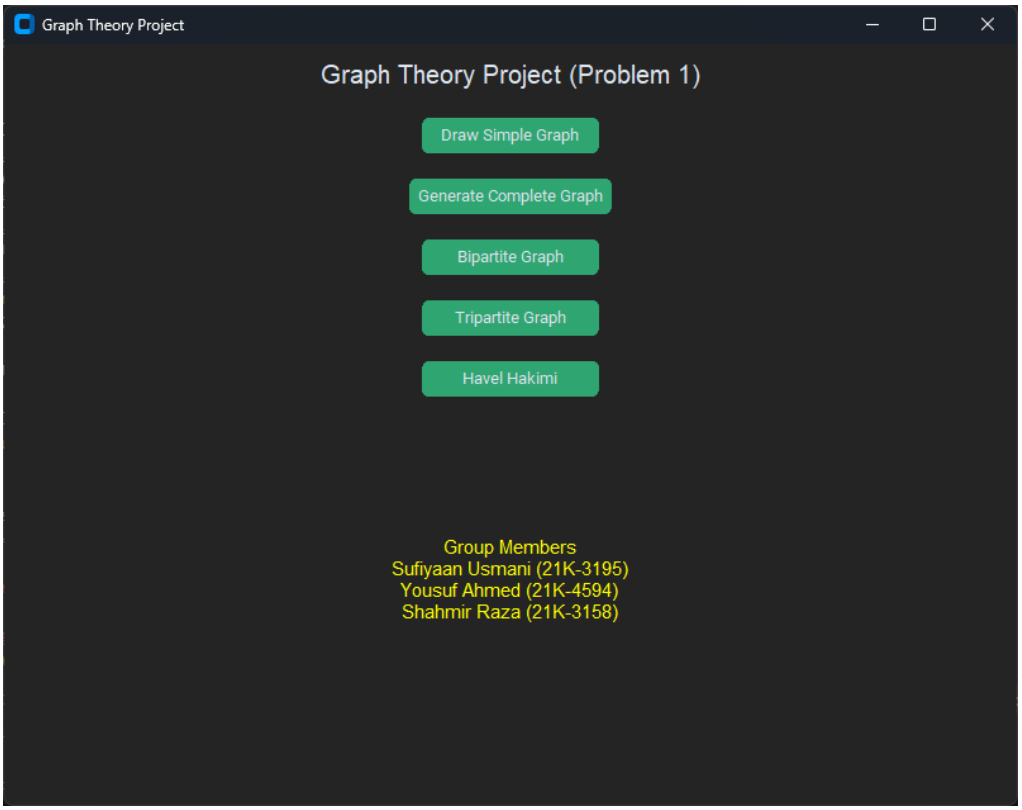
Graph theory serves as a fundamental framework with diverse applications across various domains, from computer science to natural sciences. This project aimed to implement fundamental graph operations through a Python-based graphical interface using **tkinter** and also details the practical application of graph theory concepts through Python implementations and the exploration of a research paper's algorithmic approach in predicting emerging health risks.

Objectives:

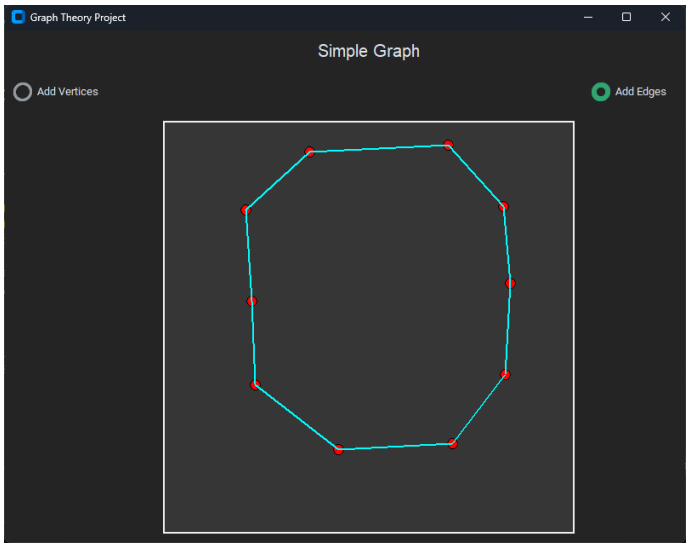
This project aimed to:

- Implement fundamental graph operations using Python.
- Review and implement an algorithm from a research paper related to predicting health risks.
- Explore alternative applications of the chosen algorithm.

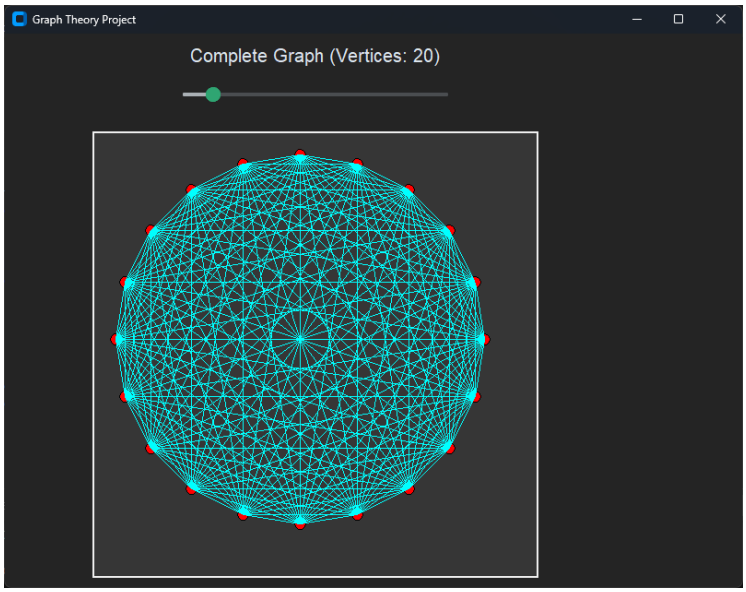
Graph Operations in Python:



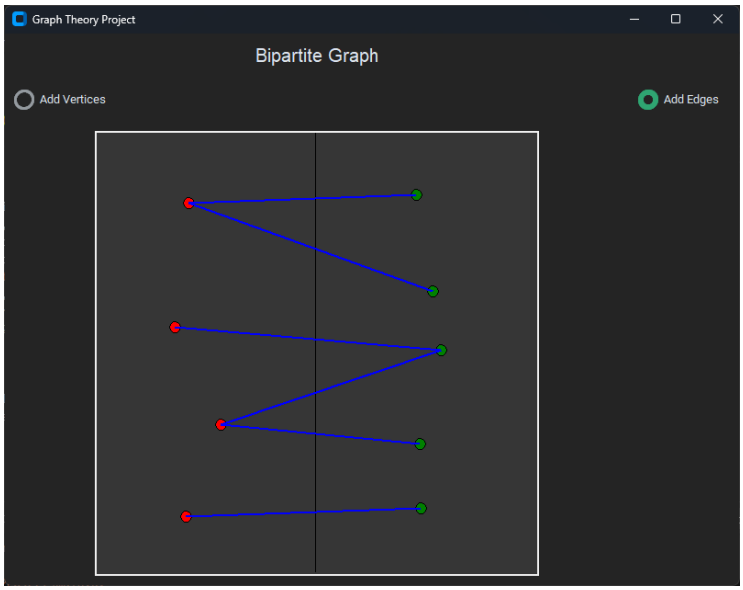
1. **Draw Simple Graph:**
   This functionality is implemented in the simpleGraph function. It first clears the content of the application window, then sets up the GUI for drawing a simple graph. The user can select whether to add vertices or edges using radio buttons. The draw_point function is used to draw vertices on the canvas when the user clicks on it, and the drawEdge function is used to draw edges between vertices.



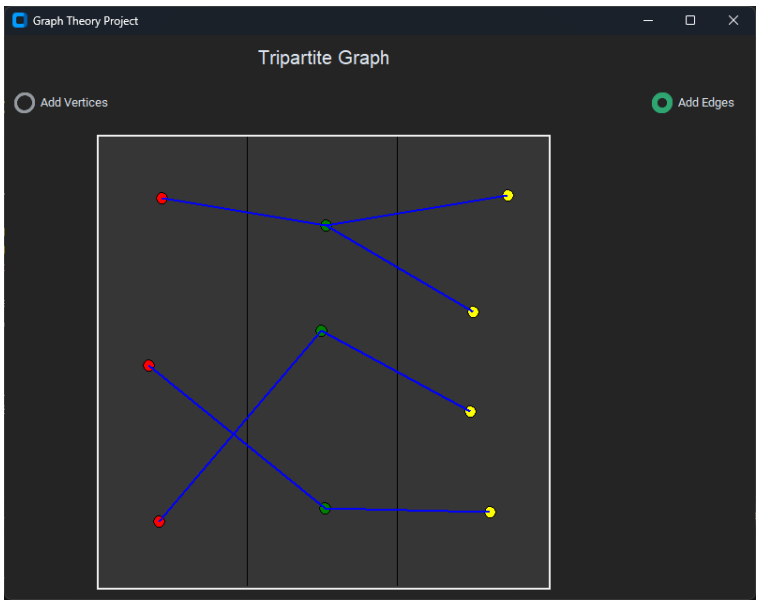
2. **Generate Complete Graph:**
   This functionality is implemented in the completeGraph and completeGraphSlider functions. The completeGraph function sets up the GUI for generating a complete graph, and the completeGraphSlider function generates the graph based on the number of vertices specified by the user. The vertices are evenly distributed around a circle, and each vertex is connected to every other vertex.



3. **Bipartite Graph:**
   This functionality is implemented in the bipartiteGraphButtonClicked, drawBipartitePoint, and drawBipartiteEdge functions. The bipartiteGraphButtonClicked function sets up the GUI for drawing a bipartite graph. The drawBipartitePoint function is used to draw vertices on the left or right half of the canvas depending on where the user clicks, and the drawBipartiteEdge function is used to draw edges between vertices of different colors.

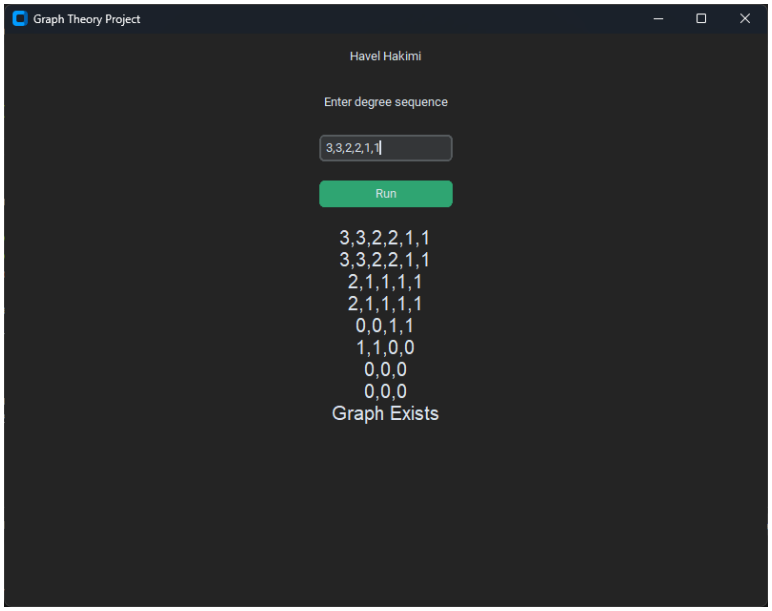


4. **Tripartite Graph:**
   This functionality is similar to the bipartite graph functionality, but it allows the user to draw a tripartite graph, which is a graph whose vertices can be divided into three disjoint sets. The relevant functions are tripartiteGraphButtonClicked, drawTripartitePoint, and drawTripartiteEdge.



5. **Havel Hakimi:**

   This functionality is implemented in the havelHakimi and havelHakimiWindow functions. The havelHakimiWindow function sets up the GUI for entering a degree sequence and running the Havel-Hakimi algorithm. The havelHakimi function implements the algorithm, which involves repeatedly sorting the degree sequence in descending order and reducing the degrees of the highest-degree vertices. The algorithm stops when all degrees are zero (in which case the degree sequence is graphical) or when it's impossible to reduce the degrees as required (in which case the degree sequence is not graphical). The steps of the algorithm are displayed in the GUI.



Research Paper Review and Implementation:

Title: Implementation of Multi-Context Mining-Based Graph Neural Network for Predicting Emerging Health Risks

Abstract:

This report presents an implementation of the method proposed in the research paper "Multi-Context Mining-Based Graph Neural Network for Predicting Emerging Health Risks". The method uses a Graph Neural Network (GNN) and multi-context mining to predict potential health risks. A simplified version of the algorithm was implemented using a Graph Neural Network library and a publicly available graph dataset.

Introduction:

The ability to predict potential health risks is crucial in the field of healthcare. With the advent of deep learning and neural networks, more sophisticated methods for health risk prediction have been developed. This report focuses on the use of Graph Neural Networks (GNNs), which are particularly suited for handling data with complex relationships and interdependencies.

Literature Review:

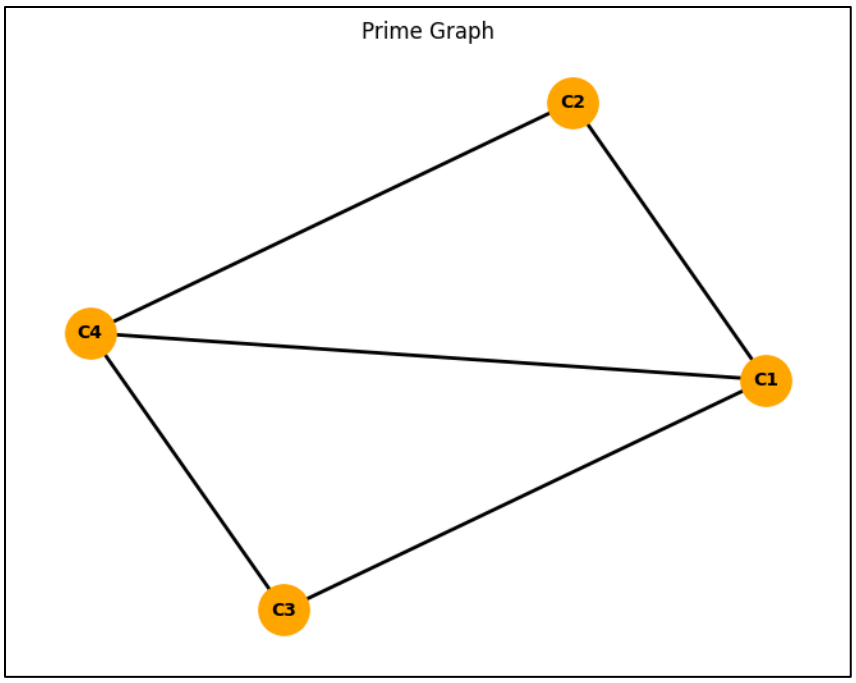
The research paper "Multi-Context Mining-Based Graph Neural Network for Predicting Emerging Health Risks" proposes a method for predicting health risks using a GNN and multi-context mining. The paper highlights the increasing prevalence of chronic diseases and the need for effective prediction methods. The proposed method involves three steps: data preprocessing, multi-context mining, and graph representation.

Methodology:

The method proposed in the research paper involves collecting and preprocessing health information, performing context mining to generate a feature map for the graph extension, and augmenting the graph with the feature map to predict latent risks. Implementing this method in Python presents challenges related to data availability, as the specific health data used in the paper may not be readily accessible.

Implementation:

A simplified version of the algorithm was implemented using the PyTorch Geometric library and a publicly available graph dataset. The implementation focused on the process of creating the graph, performing feature extraction and selection as a simplified form of context mining, and training the GNN for a prediction task.



Conclusion:

This report presented an implementation of the method proposed in the research paper "Multi-Context Mining-Based Graph Neural Network for Predicting Emerging Health Risks". The implementation demonstrated the potential of GNNs for predicting health risks and highlighted the challenges related to data availability.

Alternate Applications:

1. **Social Network Analysis:**

   GNNs are used to analyse social networks by modelling the relationships between users. They can predict user behaviour, identify influential users, and detect communities within the network.

2. **Recommendation Systems:**

   GNNs can be used in recommendation systems to model the interactions between users and items. They can predict user preferences and recommend items based on these predictions.

3. **Chemoinformatics:**

   GNNs are used in chemoinformatics to predict the properties of molecules. They model the atoms in a molecule as nodes in a graph and the bonds between atoms as edges.

References:

1. J.-W. Baek, K. Chung, "Multi-Context Mining-Based Graph Neural Network for Predicting Emerging Health Risks", IEEE, 2023.
2. Monti, F., Boscaini, D., Masci, J., Rodola, E., Svoboda, J., & Bronstein, M. M. (2017). Geometric deep learning on graphs and manifolds using mixture model CNNs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5115-5124).
3. Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L., & Leskovec, J. (2018). Graph convolutional neural networks for web-scale recommender systems. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (pp. 974-983).
4. Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural message passing for quantum chemistry. In Proceedings of the 34th International Conference on Machine Learning-Volume 70 (pp. 1263-1272). JMLR. org.