**OPEN ENDED**

**ANALYSIS AND DESIGN OF ALGORITHMS CSE(303)**

**A logo of a university

Description automatically generated**

In partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY

AMITY UNIVERSITY, UTTAR PRADESH

Submitted By: Submitted to:

**Anant Singh Dr. Garima Aggarwal**

**(A2305222216)**

**5CSE3-Y**

# OPEN ENDED

**AIM:** Arbitrage detection in financial graphs using Bellman Ford.

**SOFTWARE USED:** Python interpreter on machine.

**THEORY:**

In my project, I implemented the Bellman-Ford algorithm to detect arbitrage opportunities in financial graphs, focusing on currency exchange. The goal is to find cases where I can make a risk-free profit by exploiting differences in currency exchange rates.

**Bellman-Ford Algorithm:** I used this algorithm because it not only computes the shortest paths between currencies but also helps detect negative weight cycles in the graph. These cycles represent arbitrage opportunities.

This is a graph-based algorithm used to compute the shortest paths from a single source vertex to all other vertices in a weighted graph. Unlike other shortest path algorithms like Dijkstra's, the Bellman-Ford algorithm can handle graphs with negative weight edges, making it particularly useful in cases where negative cycles (cycles that reduce the total path cost) exist.

**Key Features:**

1. **Single-source shortest path**: The algorithm starts from a single source vertex and calculates the shortest path to every other vertex in the graph.
2. **Works with negative weights**: Unlike Dijkstra’s algorithm, Bellman-Ford can handle negative edge weights, which is essential for detecting certain types of problems, such as arbitrage in financial graphs.
3. **Cycle detection**: The algorithm can detect negative weight cycles (a cycle in the graph where the sum of the weights is negative). In financial contexts, these represent arbitrage opportunities.

**Financial Graphs:** In my graph, each node represents a currency, and the edges between them represent exchange rates. The algorithm checks for cycles where the product of exchange rates across multiple currencies results in a value greater than 1, indicating arbitrage.

**ARBITRAGE:** This is the practice of exploiting price differences between markets to make a risk-free profit. In currency exchange, it involves buying and selling currencies across different markets where exchange rates are inconsistent, allowing traders to profit from those discrepancies.

**WHY USE BELLMAN FORD:** I chose the Bellman-Ford algorithm for my project because it can handle negative weight edges in a graph, which is essential for detecting arbitrage opportunities in financial markets. Other algorithms, like Dijkstra's, cannot process negative weights and would miss detecting negative cycles, which represent potential arbitrage. Bellman-Ford’s ability to identify negative cycles makes it perfect for this use case, where these cycles reveal profitable currency exchange routes.

**BACKEND:**

**A computer code on a black background

Description automatically generated**

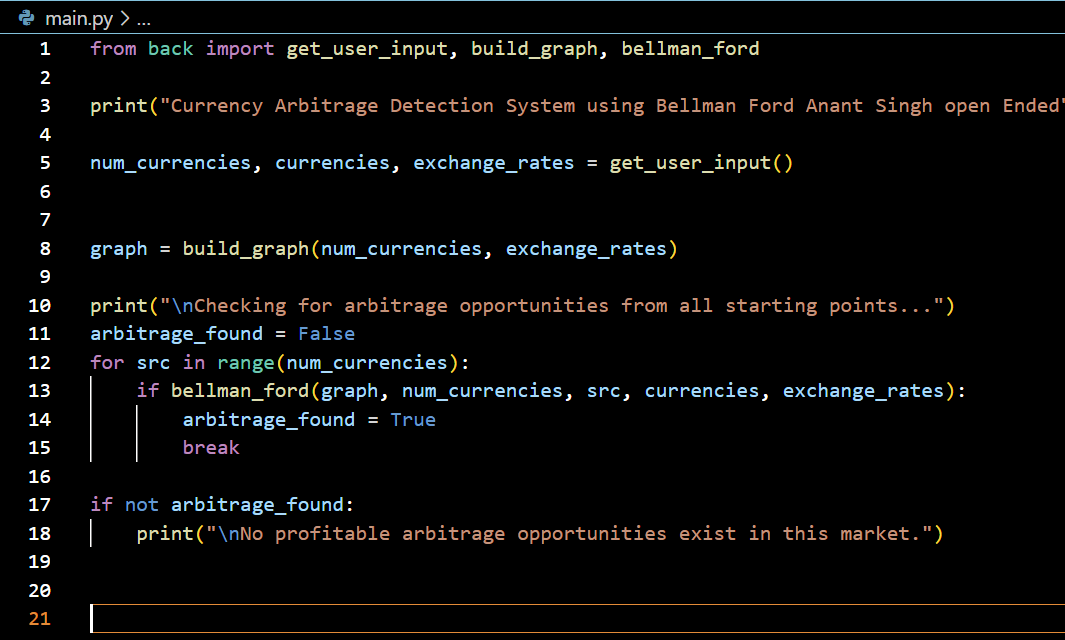
**A black screen with text on it

Description automatically generated**

**A screen shot of a computer

Description automatically generated**

**Main program:**

****

**OUTPUT:**

**ARBITRAGE NOT FOUND:**

A screenshot of a computer screen

Description automatically generated

**ARBITRAGE FOUND:**

A screenshot of a computer

Description automatically generated

**CONCLUSION:** In conclusion, this project successfully applies the Bellman-Ford algorithm to detect arbitrage opportunities in currency exchange markets. By leveraging the algorithm’s ability to handle negative weight cycles, we can identify profitable trading routes based on discrepancies in exchange rates. This project demonstrates how algorithmic approaches can be used in real-world financial applications.