Emad Hassan
CMPSC 463Section 001
Dsgn Analysis Algs
Pr. Janghoon Yang

1. Description of the Project

Overview

The project is designed to analyze large-scale financial datasets (such as stock prices, transaction logs, or cryptocurrency data) to detect patterns, trends, and anomalies. The primary problem addressed is the need for efficient processing of these large datasets to help users make data-driven financial decisions. The system uses **divide-and-conquer algorithms** to ensure that the analysis can handle the complexity and size of financial datasets while maintaining performance.

Goals of the Analysis

The main goals of the analysis are:

- To detect trends in financial data, such as identifying periods of maximum gain or loss.
- To sort financial data by time for further trend analysis.
- To detect anomalies in the data, such as irregular spikes or dips in stock prices or transaction volumes, which may signal financial irregularities or opportunities.

Type-Specific Considerations

- **Financial Data**: For this project, financial datasets such as stock prices or cryptocurrency data are ideal due to their time-series nature, which allows for meaningful trend and anomaly detection.
- **Algorithms Used**:
- **Merge Sort**: Efficiently sorts large financial datasets by time.
- **Kadane's Algorithm**: Identifies the period of maximum gain or loss.
- **Closest Pair of Points Algorithm**: Detects anomalies in financial data based on price deviations.

2. Structure of the Code with Diagram and Comments

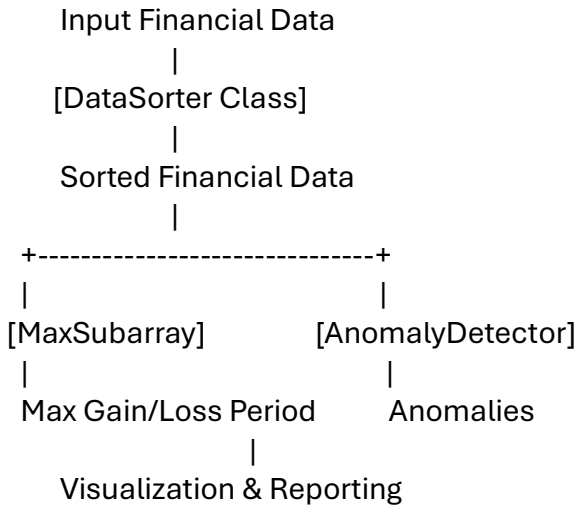
Code Structure

The system is modular and split into three primary classes:

1. **DataSorter**: Implements the merge sort algorithm for time-series data sorting.
2. **MaxSubarray**: Implements Kadane's algorithm for detecting the maximum gain/loss period.
3. **AnomalyDetector**: Uses the closest pair of points algorithm to identify anomalies in the dataset.

Block Diagram

Here's a basic block diagram showing how the components interact:



Class Summaries

- **DataSorter**:
- **Purpose**: Sorts financial data by timestamp using merge sort.
- **Key Method**: merge_sort(data): Recursively splits and sorts data.
- **MaxSubarray**:
- **Purpose**: Finds the period of maximum gain or loss in a time-series dataset.
- **Key Method**: find_max_gain(data): Implements Kadane's algorithm to detect the highest gain/loss period.
- **AnomalyDetector**:

- **Purpose**: Detects anomalies in financial data using the closest pair of points algorithm.
- **Key Method**: closest_pair(data): Sorts data by prices and identifies the closest price pairs to detect anomalies.

3. Instructions on How to Use the System

Steps to Run the System

1. **Load Data**: Prepare the financial dataset in the format of a list of tuples, where each tuple contains a timestamp and a corresponding price (or transaction volume).
2. **Install Dependencies**: Ensure that the required libraries (e.g., matplotlib) are installed using:
pip install -r requirements.txt
3. **Run the Script**: Execute the main.py script:
python main.py
4. **Output**: The system will process the data and generate visualizations that show:
   - The stock price trends over time.
   - The period of maximum gain or loss.
   - Any detected anomalies (price spikes or dips).

Generating Reports

- The system will automatically generate a visualization that highlights key financial insights, which can be used for reporting purposes.

4. Verification of Code Functionality

Toy Example

A toy dataset was used for verification, consisting of 50 timestamps and corresponding stock prices. The system successfully sorted the dataset, identified the maximum gain period, and detected anomalies.

Example Scenario

Sample data:
[(1, 150), (2, 180), (3, 130), (4, 210), ...]

- **Max Gain/Loss Period**: The system identified the time period where stock prices experienced the highest increase (max gain).
- **Anomalies Detected**: The system flagged sudden price dips or spikes as potential anomalies.

Screenshots:

Below is a screenshot showing the system in action, where the trend and anomalies are highlighted.

5. Discussion of Findings

Insights Gained

- **Trends**: The system successfully detected periods of maximum gain, which is useful for identifying stock performance trends over time.
- **Anomalies**: The closest pair of points algorithm detected anomalies in the data, such as unexpected price drops or spikes. This can assist in identifying potential fraud or investment opportunities.

Challenges Faced

- **Handling Large Datasets**: Although the divide-and-conquer approach offers scalability, handling extremely large datasets (e.g., millions of transactions) can still be a challenge. Future work may require optimizing memory usage or applying parallel processing techniques.

Limitations

- **Closest Pair in 1D**: The current implementation of the closest pair of points works only in 1D (price vs. time). Extending this to handle multi-dimensional data (e.g., comparing multiple stocks) would provide deeper insights into market behavior.

Suggested Improvements

- **Multi-Dimensional Anomaly Detection**: Expanding the anomaly detection algorithm to work in multiple dimensions (e.g., comparing two stock prices simultaneously) could enhance the system's capabilities.
- **Performance Optimization**: Further optimization can be done to handle extremely large datasets more efficiently, potentially leveraging parallel computing or cloud-based processing.