Technical Report: Final Project
EECE 2560: Fundamentals of Engineering Algorithms

Noah Assam, Chance Bowman
Department of Electrical and Computer Engineering
Northeastern University
assam.n@northeastern.edu, bowman.c@northeastern.edu

November 18, 2024

Contents

1 Project Scope

This project aims to develop a stock trading simulator that leverages greedy algorithms and historical data analytics to enable users to make profitable trading decisions, in real-time. The system will address the challenge of making trading decisions in a volatile stock market. It provides data-driven trade suggestions, as well as a user-friendly interface, for easier interaction with stock market trends.

The project's main objectives are:

- To develop a user interface for real-time stock trading.
- To implement, and refine, greedy algorithms that optimize trading decisions.
- To use historical stock data to verify algorithmic accuracy and refine strategies.

The expected outcomes include a functioning web-based system, proper documentation, and a final project report.

2 Project Plan

2.1 Timeline

The overall timeline for the project is divided into phases:

- **Week 1 (October 7 - October 13)**: Define project scope, establish team roles, and outline skills/tools.
- **Week 2 (October 14 - October 20)**: Research and design system architecture.
- **Week 3 (October 21 - October 27)**: Begin UI development, greedy algorithm prototyping, and some initial tests with historical data.
- **Week 4 (October 28 - November 3)**: Further refine greedy algorithm, implement a more advanced trade logic, and begin backend integration.
- **Week 5 (November 4 - November 10)**: Develop graph algorithms for trend analysis. Complete database integration.
- **Week 6 (November 11 - November 17)**: Testing and debugging.
- **Week 7 (November 18 - November 28)**: Final presentation, report submission, and project closure.

2.2 Milestones

Key milestones include:

- Completion of architecture design (End of Week 2).
- User interface prototype (End of Week 3).
- Greedy algorithms competition (End of Week 5).
- Graph algorithms completion (End of Week 7).
- Final testing and debugging (End of Week 8).

3 Team Roles

- **Team Member 1:** Responsible for user interface and database development.
- **Team Member 2:** Responsible for backend development and documentation.

These roles are flexible and may change as the project progresses.

4 Methodology

4.1 Pseudocode and Complexity Analysis

Provide pseudocode for the primary algorithm(s) implemented in the project. Explain each step in detail and analyze the time and space complexity for each algorithm, highlighting its efficiency and suitability for the project requirements.

TODO

Notes:

- Primary algorithms: greedy, graph
  - generate.trade.signal, *TODO*
  - generate.trade.signal: if short term moving average (STMA) is above long term moving average (LTMA), and current price is greater than STMA, buy. if STMA is below LTMA and current price is less than STMA, sell. otherwise, neutral reaction.
  - *TODO*: *TODO*
  - generate.trade.signal: generally, O(n); if only a single holding, O(1)
  - *TODO*: *TODO*

4.2 Data Collection and Preprocessing

Explain how the data was collected and describe the preprocessing steps necessary for the algorithm. *TODO*

Notes:

- Data to be retrieved from yfinance library, making use of real-time stock values
- Performance data and trade history to be stored from collected user inputs

5 Results

Present the results, including key performance metrics and visualizations (e.g., tables, charts) that illustrate the effectiveness of the algorithms. *TODO*

6 Discussion

Interpret the results and discuss their implications. Compare the findings with the initial objectives, and highlight any discrepancies or unexpected outcomes. *TODO*

7 Conclusion

Summarize the key findings, discuss limitations, and suggest areas for future improvement. *TODO*

8 References

Include all sources cited in the report. *TODO*

A Appendix A: Code

Include relevant code sections here. *TODO–UPDATE*

B Appendix B: Additional Figures

Provide any additional figures or tables that support the analysis. *TODO*

C Appendix C: Iteration Updates

C.0.1 Iteration 2 (October 10)

The team successfully began the initial setup of the development environment and GitHub repository. Research on backend development has begun. The team has encountered issues with the plan for constructing the greedy algorithms. Will resolve by reviewing class resources and utilizing tutorials.

C.0.2 Iteration 3 + 4 (October 27)

Greedy Algorithm Development

Following the team's initial challenges, we implemented a foundational version of the greedy algorithm. It focuses on optimizing buying and selling actions, based on trend indicators. Such an iterative approach has allowed for the start of a basic setup, enabling a gradual improvement of the algorithm by:

- Introducing technical indicators, such as short- and long-term moving averages.
- Integrating a volatility check such as to minimize risk.
- Testing with historical stock data to evaluate decision-making accuracy.

Next steps involve the refinement of this algorithm, as well as testing against edge cases to ensure proper performance.

Evidence of code execution may be viewed in the Appendix, as Fig. 1, Fig. 2, Fig. 3. Evidence of output may be viewed as Fig. 4.

User Interface Development

The team has developed a relatively intuitive interface for the simulator, enabling users to interact with real-time stock trends and make guided trading decisions. Currently, the interface includes:

- Real-time visualization of stock trends.
- Simple trade actions (buy, sell, hold), which are based on algorithmic suggestions.
- A dashboard displaying portfolio balance, stock holdings, and a trade history.

Future iterations will enhance usability, possibly adding features for a deeper interaction with market data.

Backend and Database Integration The backend development is underway, though more central efforts will be completed in the upcoming weeks.

Algorithm Testing

Early testing has involved simulated datasets to verify some basic functionalities. Such has included:

- The testing of market indicators and market trend analysis.
- The testing of trade signal generation and execution.
- The testing of invalid trades, (e.g., selling unowned holdings, buying holdings without sufficient stock).

Evidence of testing may be viewed in the Appendix, as Fig. 5, Fig. 6.

System Architecture
The system architecture follows a layered approach:

- *User Interface:* This layer allows users to interact with the simulator. It includes a graphical interface displaying stock trends, trade options, and user account details.
- *Backend Services:* This layer is responsible for processing user requests, managing trade logic, and handling data communications. It includes:
  - Greedy Algorithm Module: This component handles trading decisions, based on implemented greedy algorithms. It continuously analyzes market data in order to optimize trade decisions.
  - Data Processing Module (TODO): This component manages the retrieval and analysis of historical stock data, utilizing SQL for database interactions.
- *Database Layer (TODO):* This layer uses SQL to store user data, historical stock information, as well as trade history.

Data Representation:
The system utilizes various data structures for data management:

- *Stock Data Representation:* Each stock is represented as an object, containing attributes such as stock symbol, current price, historical prices, and trading volume.
- *User Portfolio:* User portfolios are stored as a collection of stock object, allowing for access to both stock holdings and performance metrics.
- *Trade History:* All trade actions are recorded in a format that includes timestamps, actions, and quantities.

New Challenges:
As of this week, there have not yet been any new challenges encountered. Such is most probably due to the relative generality of the current algorithm and lack of a backend. Ergo, no new solutions are proposed. This will be kept updated for future iterations.

C.0.3 Iteration 5 (November 10)

Plans Made:

Team Member 1:

- Unit Testing
  - Implement edge case scenarios and stress testing
  - Create error handling
  - Design automated validation protocols for consistency
- New Analysis Algorithm
  - Develop implementation
  - Create integration with existing systems
- Initial Presentation
  - Create system diagrams and performance metrics
  - Outline implementation progress and milestones
  - Draft initial findings and technical achievements

Team Member 2:

- Storage Improvements
  - Implement data compression strategies
  - Optimize database operations and indexing
  - Design efficient caching mechanisms
- Runtime Enhancement
  - Implement processing optimizations
  - Optimize memory usage patterns
- Final Presentation Refinement
  - Content development
  - Create data visualizations and demonstrations
  - Prepare summary and key messages