ESE 4970(4971) Capstone Design Project Final Report

Submitted to the Department of Electrical and Systems Engineering

Washington University in St. Louis

Design and Evaluation of a Market Integrity Analysis System for Prediction Markets

Time Period: January 2026 — May 2026

Date of Submission: May 1, 2026

Name	Department	Degree	Email
Al Malik Anand	ESE	B.S. SSE	a.malikanand@wustl.edu
Nick Haught	ESE	B.S. EE	haught@wustl.edu

Role	Information
Project Client	Prof. Yevgeniy Vorobeychik
Client Affiliation	Dept. of Computer Science & Engineering, Washington University in St. Louis
Client Email	yvorobeychik@wustl.edu
Faculty Instructor	Prof. Ben Wormleighton
Instructor Affiliation	Washington University in St. Louis
Instructor Email	bwormleighton@wustl.edu

This report is submitted in partial fulfillment of ESE 4970/4971 Capstone Design Project.

Executive Summary

Objective

This project studies market integrity in native asset markets, with a primary focus on prediction markets, through a structured simulation and detection framework grounded in market microstructure and optimization. The project follows multiple levels of analysis, beginning with anti manipulation frameworks in pre-existing financial markets, and seeks to transfer, adapt, and extend these mechanisms toward establishing a viable integrity checking system within an emerging and relatively under regulated asset class. The work treats market integrity as an engineering design problem, where detection, response, and structural robustness must be embedded directly into the market system.

"What are the most efficient ways of designing mechanisms that can detect and constrain manipulation attempts in native asset markets, particularly in low liquidity prediction environments?"

Market Structure and Context

Prediction markets operate through financial instruments known as event contracts, which are structured as binary or discrete outcome securities tied to real world events. These contracts trade between zero and one, where price reflects the implied probability of the event occurring through a wisdom of the crowds aggregation process. The reliability of this mechanism depends on sufficient liquidity, balanced participation, and resistance to strategic interference. In practice, these conditions are often not met, particularly in early stage or thinly traded markets.

Methodological Approach

To analyze these dynamics, a modular agent based system is developed using the `Mesa` framework, structured around a continuous double auction with a central limit order book. The system supports heterogeneous agents including noise traders, informed traders, and multiple classes of strategic manipulators. These include cyclic wash trading agents, coordinated ring based manipulation structures, and large capital whale regimes. The simulation environment is designed to isolate and stress test specific manipulation behaviors under controlled market conditions, particularly in low liquidity settings.

A PCA based investigative framework was used at the early stage of the project to guide scope and identify broad categories of manipulation behavior. This step was not intended as a primary result, but rather as a structuring tool to distinguish between quote based distortion mechanisms and volume based manipulation mechanisms, which informed the design of the simulation and detection framework.

The central methodological focus is the analysis of wash trading detection under a fixed structural formulation, comparing Dynamic Programming, Integer Linear Programming, and Branch and Bound approaches. The results indicate that Branch and Bound achieves higher detection accuracy and computational efficiency relative to the other methods within the simulated environment. This performance advantage is largely driven by the pruning mechanism inherent to Branch and Bound, which allows large portions of the solution space to be eliminated early based on bounding conditions.

However, this efficiency is closely tied to the structure of the simulated problem. In the designed environment, trade flows and volume constraints are sufficiently well defined such that pruning is effective. In real world markets, where multiple combinations of trades can satisfy similar volume conditions and where structural ambiguity is higher, the pruning advantage may weaken. In such cases, the solution space becomes

flatter and more symmetric, reducing the effectiveness of bounding and potentially limiting the scalability advantage of Branch and Bound relative to alternative methods.

Core Contribution

The central technical contribution is the reformulation of a digraph based wash trading detection problem into a constrained optimization framework. Prior approaches model wash trading using directed graph representations of trade flows and solve the resulting subset matching problem using dynamic programming techniques. This project extends that formulation by introducing Integer Linear Programming and Branch and Bound methods applied to the same structural problem. This enables the detection task to be expressed explicitly as an optimization problem over trade flow constraints and allows direct comparison between solution paradigms in terms of computational efficiency and detection performance.

Key Findings

The results indicate a relatively balanced dominance between quote based distortion and volume based manipulation in low liquidity markets, particularly within fintech native crypto environments and event contract platforms. This suggests that both price signaling and liquidity representation can be systematically distorted under realistic conditions.

Temporal analysis shows that whale induced distortions produce measurable transient responses in price and liquidity, while stress testing reveals that adversarial strategies can evade purely structural detection methods by satisfying graph based constraints while still distorting outcomes. These findings highlight the limitations of single method detection systems.

Conclusion

This project demonstrates that market integrity in prediction markets must be addressed through integrated system design. Effective detection requires the combination of structural, temporal, and behavioral approaches within a unified framework. The results provide a foundation for developing scalable integrity mechanisms tailored to emerging financial systems.

1. Background and Justification

1.1. Prediction Markets and Informational Role

Prediction markets have emerged as a mechanism for aggregating dispersed information into a single market signal through trading activity. These markets operate through event contracts, where prices are interpreted as implied probabilities of future outcomes. In theory, this structure allows a wide range of participants, each with partial or private information, to contribute to a collective forecast through price formation.

Prior work has shown that prediction markets can perform competitively with, and in some cases outperform, traditional forecasting methods such as polling or expert analysis. The paper by Rothschild and Sethi showed that even in the presence of biased traders and attempted manipulation, prediction markets were able to produce relatively accurate forecasts due to the interaction of heterogeneous trading strategies.

More recent large scale experimental work has reinforced the idea that these markets act as powerful information aggregators, often providing probability estimates in domains where no alternative structured forecasting tools exist. However, this same feature introduces a critical vulnerability. Because these probabilities are observable and widely interpreted as signals of likelihood, they extend beyond purely financial relevance and begin to influence perception, belief formation, and decision making.

In practice, implied probabilities from prediction markets are often referenced informally in media, policy discussions, and strategic decision making environments. These signals can influence expectations about geopolitical events, economic outcomes, and electoral processes. This creates a feedback loop in which market prices do not simply reflect beliefs, but can also shape them.

1.2. Manipulation Risk and Systemic Implications

The ability of prediction markets to act as informational signals introduces strong incentives for manipulation. If market prices are interpreted as probabilities by external observers, then shifting those prices can alter perception even without changing underlying fundamentals.

The paper by Rasooly and Rozzi demonstrated through a large scale field experiment that prediction markets are indeed manipulable. Their results showed that even relatively small trades could produce price effects that persist over extended periods, in some cases lasting up to 60 days.

This persistence is particularly concerning in low liquidity environments, where the number of participants and volume of trades are limited. In such settings, fewer counteracting trades exist to correct distortions, allowing manipulative behavior to have a stronger and longer lasting effect. At the same time, these are often the environments in which prediction markets are growing most rapidly, particularly in fintech native crypto ecosystems.

More broadly, literature on market manipulation detection has consistently emphasized that trade based manipulation remains one of the most difficult forms to identify. The review by Khodabandehlou and Hashemi Golpayegani highlights that trade based manipulation operates within the normal mechanics of the market, making it difficult to distinguish from legitimate activity while still producing artificial price and volume signals.

These dynamics create a structural vulnerability. If prediction markets are used as informal signals for decision making, and those signals can be manipulated at relatively low cost, then the integrity of the market

becomes a system level concern rather than a purely financial one.

1.3. PCA Based Scope Identification and Problem Selection

At the initial stage of this project, a PCA based investigative approach was used to examine broad categories of manipulation behavior across simulated and theoretical regimes. This analysis was not intended as a primary contribution, but rather as a structuring mechanism to identify which manipulation types were most relevant within the context of prediction markets.

The PCA decomposition indicated two dominant axes of manipulation behavior. The first corresponded to quote based distortion mechanisms, including spoofing and order book manipulation, which affect price signals through transient order placement. The second corresponded to volume based manipulation mechanisms, including wash trading and coordinated trade structures, which affect perceived liquidity and transaction flow.

Within this classification, wash trading emerged as a particularly suitable focus for detailed analysis. This is due to three key factors. First, wash trading operates entirely within executed trade data, making it structurally harder to detect than quote based manipulation. Second, it directly impacts volume signals, which are often interpreted as indicators of market activity and confidence. Third, it can be represented naturally as a graph structured problem, allowing for formal optimization based detection approaches.

1.4. Justification for Wash Trading Focus

Trade based manipulation, and specifically wash trading, represents a critical gap between theoretical detectability and practical enforcement. While quote based manipulation leaves observable patterns in order book behavior, wash trading produces legitimate appearing trades that satisfy all exchange level constraints.

As highlighted in prior literature, these forms of manipulation are particularly challenging because they do not violate explicit rules in a detectable way, but instead exploit the structure of the market itself. This makes them ideal candidates for reformulation as optimization problems, where detection depends on identifying constrained patterns within trade networks rather than simple rule violations.

From an engineering perspective, this aligns directly with the design of the simulation system used in this project. Trade flows can be represented as directed graphs, allowing wash trading cycles and coordinated structures to be analyzed through combinatorial optimization methods. This provides a clear pathway for comparing different detection paradigms under controlled conditions.

1.5. Broader Justification

The motivation for this work extends beyond the detection of individual manipulation strategies. As prediction markets continue to expand and integrate into broader financial and informational ecosystems, their role as probability generating mechanisms becomes increasingly significant.

If these markets are to be used, even informally, as signals for policy, strategy, or public interpretation, then their integrity must be treated as a system level design problem. This project addresses that need by combining simulation, structural modeling, and optimization based detection to better understand both the vulnerabilities and the limits of current approaches.

In this context, the focus on wash trading is not only a technical choice, but a foundational step toward building more robust and scalable integrity mechanisms for emerging market systems.

2. Methods

2.1. Methodological Overview

The methodological framework is designed as a modular system that integrates controlled simulation, transaction level data construction, and optimization based detection. Rather than treating manipulation as a statistical anomaly, the approach models it as a structured interaction within a market system, where both legitimate and adversarial behaviors are explicitly generated and observed.

At a high level, the system follows a three stage pipeline. A synthetic market environment produces transaction level data under controlled conditions. This data is then transformed into candidate trade relationships by identifying structurally consistent buy sell interactions. These candidates are subsequently evaluated using constrained optimization methods, where detection is framed as selecting globally consistent patterns rather than isolated anomalies.

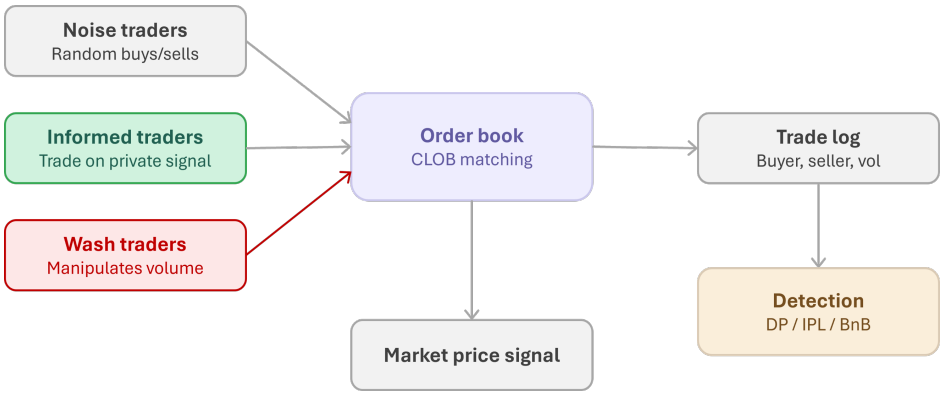

Figure 1: Simulation architecture. Agents submit orders to the CLOB, matched trades are logged, and detection algorithms scan the trade log for wash ring cycles and anomalous patterns.

The system operates entirely at the transaction level, using price, volume, time, and agent identity to construct a detailed representation of market activity. This enables the identification of manipulation as a structural property of trade interactions rather than a feature of individual trades. In particular, the framework adopts the characterization that wash trades emerge from tightly matched orders and closed transaction structures, an interpretation developed in prior work and used here as the conceptual foundation for detection.

Candidate trade pairs are constructed by matching orders based on proximity in time and price, along with near equivalence in volume. These relationships form a weighted structure over which detection is performed. The objective is then defined over this structure, selecting subsets of interactions that satisfy global consistency constraints such as volume balance and exclusivity of participation.

A key feature of the design is that ground truth manipulation patterns are embedded directly within the simulation. This allows for controlled evaluation of detection performance under varying levels of noise, structural ambiguity, and adversarial behavior.

The architecture is intentionally modular, allowing the simulation environment, manipulation behaviors, and detection framework to be developed and analyzed independently. This separation enables systematic evaluation of how specific manipulation structures interact with detection mechanisms, while maintaining a consistent underlying system representation.

2.2. Agent-Based Market Simulation

The market environment is implemented as an agent based simulation using the `Mesa` framework, designed to replicate a continuous double auction (CDA) with a central limit order book (CLOB). The objective of the simulation is not to reproduce full market realism, but to construct a controlled environment in which trading behaviors, execution dynamics, and manipulation strategies can be explicitly modeled and observed at the transaction level.

The core of the market mechanism is the CLOB, which maintains two ordered collections of outstanding limit orders: bids and asks. Buy orders are stored in descending order of price, while sell orders are stored in ascending order, ensuring that the highest bid and lowest ask are always accessible. Within each price level, orders are processed according to time priority, reflecting the standard price time matching rule used in most exchange markets.

The matching engine follows the continuous double auction structure, where incoming orders are immediately checked against the opposite side of the book for execution. A trade occurs whenever a buy order meets a sell order at compatible prices. If multiple orders are eligible for execution, the earliest submitted order at the best available price is matched first. Partial fills are supported, allowing large orders to be executed across multiple counterparties. Any unmatched portion of an order is placed into the order book as a resting limit order.

Each simulation step consists of agents submitting orders to the market based on their respective strategies. The system supports multiple agent classes, including noise traders, informed traders, and structured manipulators. Noise traders generate stochastic order flow with bounded inventory and random price selection within a defined spread. Informed traders act based on a latent signal representing an underlying fair value, submitting orders with some delay and risk adjustment. These baseline agents provide a background of legitimate trading activity against which manipulation strategies can be evaluated.

The simulation explicitly tracks all executed trades, including price, volume, timestamp, and participating agents. This transaction log serves as the primary data source for downstream detection analysis. In addition, key market state variables such as best bid, best ask, spread, and last traded price are updated in real time, allowing the system to capture evolving market conditions.

The design emphasizes deterministic control over market structure while allowing stochastic variation in agent behavior. Parameters such as order arrival rates, price bounds, and inventory constraints can be adjusted to simulate different liquidity regimes. This is particularly important for studying manipulation in low liquidity environments, where the absence of sufficient counteracting order flow amplifies the impact of strategic trading behavior.

Overall, the simulation provides a flexible and extensible environment for generating structured trading data under controlled conditions. By combining a CDA based matching engine with heterogeneous agent behavior, the system enables direct observation of how different trading strategies interact within the order book, forming the foundation for the manipulation regimes and detection framework developed in subsequent

sections.

2.3. Manipulation Regimes

The simulation models two primary manipulation strategies, each producing a distinct structural signature in the trade flow data.

Wash Trading operates through cyclic self-matching, where a set of colluding agents trade among themselves in a closed loop. In the simplest case, agent A sells to agent B, who then sells back to A, inflating volume without any real transfer of ownership. A more complex variant involves coordinated ring structures, where multiple agents form a directed cycle (e.g., A $\rightarrow$ B $\rightarrow$ C $\rightarrow$ A). In this setting, no individual transaction appears suspicious in isolation, but the aggregate structure forms a closed flow of trades. These patterns manifest as strongly connected components in the directed trade graph, making them structurally identifiable but non-trivial to detect at the local level. Figure 2 illustrates the distinction between simple two-agent cycles and coordinated multi-agent rings.

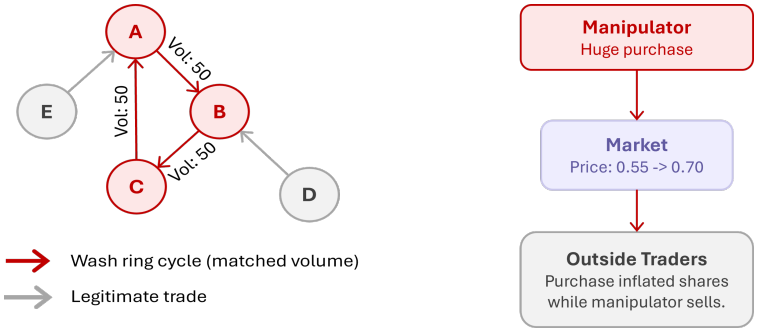

Figure 2: Digraph representations of manipulation types. Left: simple wash trade cycle between two agents. Right: a coordinated multi-agent trading ring forming a closed directed cycle.

Whale Distortion (pump and dump) is modeled through a high-capital agent capable of significantly influencing market prices. The agent initiates a sequence of aggressive buy orders, driving the mid-price upward in a short time window. This is followed by a reversal phase, where the agent exits positions through sell orders once other agents have responded to the artificially elevated price. The resulting price trajectory exhibits a three-phase pattern: an initial pump phase, a rapid dump phase, and a recovery phase where prices gradually revert toward underlying fair value.

Unlike wash trading, which primarily affects volume and trade structure, whale based manipulation produces a transient shock in price dynamics. These distortions are short-lived but measurable, and can be analyzed using response based metrics that capture deviations in price and liquidity over time. This distinction is important, as it separates structural manipulation detectable through trade relationships from temporal manipulation observable in market state evolution.

2.4. Detection Framework

2.4.1. Digraph-Based Detection with Dynamic Programming

The detection framework is built on a digraph representation of trade flows, where nodes correspond to traders and directed edges represent executed transactions. This characterization of wash trading as a graph

structured phenomenon is developed in prior work, where collusive trading activity is shown to manifest as closed transaction cycles with no net change in ownership.

In that formulation, wash trading is defined through two key structural properties. First, trades must occur in tightly matched pairs or groups, satisfying constraints on time proximity, executable price, and near equal volume. Second, these matched trades must collectively form a closed cycle in the directed graph, ensuring that the total signed volume across all participating agents sums to zero. As shown in their construction, this implies that no trader in the cycle experiences a net position change, even though transaction volume is artificially inflated.

The detection problem is therefore split into two stages. The first stage identifies candidate matched trades by scanning recent order flow within a time window and filtering based on price and volume compatibility. This effectively constructs a set of feasible trade relationships that could participate in wash activity. The second stage operates over these candidates, searching for subsets that satisfy the global zero flow constraint.

This subset selection problem is equivalent to a knapsack type formulation. Given a set of trades with associated volumes, the objective is to identify subsets whose total signed volume is approximately zero within a small tolerance. Dynamic programming is used as the baseline solution, recursively constructing feasible subsets by considering inclusion or exclusion of each trade. While this guarantees completeness, the number of possible subsets grows combinatorially, making the method computationally expensive as the number of candidate trades increases.

2.4.2. Integer Linear Programming (ILP) Formulation

Building on this formulation, the detection problem is re-expressed as an Integer Linear Program. Each candidate trade i is associated with a binary decision variable $x_i \in \{0, 1\}$ indicating whether the trade is selected as part of a potential wash structure.

The objective function maximizes the total matched volume across selected trades:

$$\max \sum_i w_i x_i$$

where w_i reflects the strength of the match, incorporating volume similarity and penalties for deviations in price and execution timing.

The core constraint enforces flow conservation across all participating agents. For each trader j, the sum of signed trade volumes must be zero:

$$\sum_{i \in \mathcal{T}_j} s_{ij} v_i x_i = 0$$

where s_{ij} encodes whether trader j is a buyer or seller in trade i, and v_i is the trade volume.

This formulation directly encodes the structural definition of wash trading derived from the digraph representation. Unlike dynamic programming, which implicitly explores feasible subsets, the ILP explicitly enforces global consistency constraints and allows the use of optimization solvers to efficiently search for valid solutions.

2.4.3. Branch and Bound Search

Branch and Bound is applied to the same subset selection problem defined in the coarse detection stage, where the objective is to identify a subset of candidate trades whose total volume matches a target order volume within a tolerance. Formally, given a set of candidate trades $\{i = 1, \ldots, N\}$ with volumes V_i, the goal is to select a subset such that

$$\left| \sum_{i \in S} V_i - V_k \right| \leq \delta_v$$

while maximizing a score function that prioritizes larger and more tightly matched trades.

This is equivalent to a knapsack type problem, where each trade represents an item with weight V_i and value w_i, and the capacity is defined by the target volume V_k. In the dynamic programming formulation, all feasible subsets are explored recursively. Branch and Bound instead traverses this same search space as a decision tree, where each level corresponds to a binary decision of including or excluding a trade.

At each node in the tree, a partial solution is defined by a subset S with accumulated volume

$$V(S) = \sum_{i \in S} V_i$$

and score

$$W(S) = \sum_{i \in S} w_i.$$

The remaining trades define a set of possible extensions to this partial solution. An upper bound on the achievable objective is computed by assuming that the remaining capacity can be filled optimally, typically by greedily adding the largest available trades without violating the volume constraint.

If this upper bound is less than the best solution found so far, the branch is pruned. This means that no extension of the current partial subset can produce a better feasible solution, and the entire subtree can be discarded without explicit enumeration. In contrast to dynamic programming, which evaluates all feasible subsets, Branch and Bound avoids exploring large portions of the search space through this bounding mechanism.

In the implemented system, this pruning is particularly effective due to the structure imposed during candidate generation. Trades are pre-filtered based on tight constraints in time, price, and volume, which significantly reduces the number of feasible combinations. As a result, many branches violate the volume constraint early or fail to improve the objective, allowing them to be eliminated quickly.

However, this efficiency is closely tied to the controlled nature of the simulation. In real market data, where multiple combinations of trades may satisfy similar volume constraints and where trade sizes are less structured, the upper bound becomes less tight. This reduces the effectiveness of pruning, as more branches remain viable for longer in the search process, increasing computational complexity.

2.5. Transient Response Analysis

When the whale agent injects a large capital shock into the market, the mid-price reacts like a dynamical system getting hit with a step disturbance. This is basically the same kind of transient response from circuits and control systems. The price overshoots its steady-state value, bounces around, and eventually settles back

within some band around the true value.

Three control theory metrics are used to measure this:

Overshoot is how far above the true value the price gets at its peak. In the whale regime, overshoot reached 31.2%, meaning the market price was temporarily way off from the actual fair value of 0.55.

Undershoot captures the drop below fair value during the dump phase. The measured undershoot was 15.9%.

Settling time is how many ticks it takes for the price to get back to and stay within a 5% band around the true value. Without any detection running, the market took over 40 ticks to settle. With BnB detection active (around 4-tick latency), settling time dropped because the manipulation got flagged before the full pump could play out.

The big takeaway here is that detection speed directly controls how much damage the manipulation does. Faster detection means a tighter feedback loop, so the price distortion stays smaller and the market recovers faster. Figure 3 shows this clearly: BnB detection (green) keeps the overshoot to 1.9%, DP detection (blue) allows 5.1%, and no detection at all (red) results in 8.1% overshoot with recovery taking over 80 ticks.

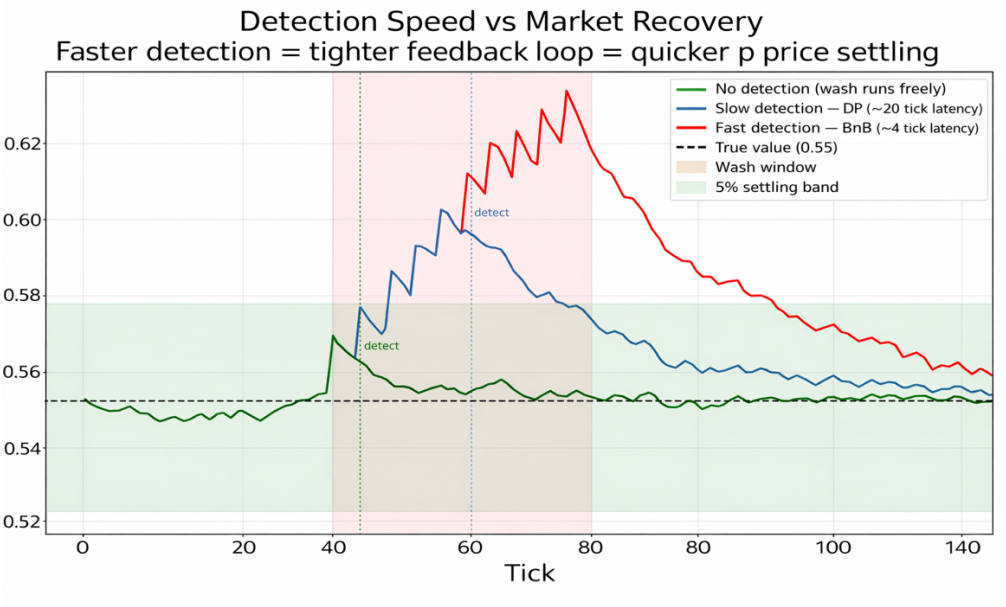

Figure 3: Detection speed vs. market recovery. Faster detection produces a tighter feedback loop and quicker price settling. The green band represents the 5% settling range around the true value of 0.55.

2.6. Success Criteria

The effectiveness of the proposed detection framework is evaluated along three primary dimensions: detection accuracy, computational performance, and market impact. These criteria are designed to jointly assess whether the system is both theoretically sound and practically deployable in real-time market environments.

Detection Accuracy. Detection performance is measured using standard classification metrics, including precision, recall, and false positive rate, computed against ground-truth labels embedded within the simulation. Precision captures the proportion of detected structures that correspond to true manipulation events, while recall measures the ability of the system to recover all known instances of manipulation. The false

positive rate is particularly important in this setting, as excessive false alarms reduce the usability of the system in practice and may flag legitimate trading activity as suspicious.

These metrics are evaluated across different manipulation regimes and market conditions, including varying liquidity levels and noise intensities. Because ground truth is explicitly controlled in the simulation, the framework allows direct comparison between detection methods, ensuring that differences in performance can be attributed to the algorithm rather than data ambiguity.

Computational Performance. Runtime performance is evaluated as a function of the number of orders and candidate trades processed. This is critical for determining whether the detection system can operate in near real-time conditions. In particular, the scalability of each method is assessed by measuring execution time across increasing order sizes, with attention to whether runtime remains within acceptable operational thresholds (e.g., sub-100 millisecond latency for moderate scale inputs).

The comparison between Dynamic Programming, Integer Linear Programming, and Branch and Bound is central to this evaluation. While Dynamic Programming provides a baseline for correctness, its exponential growth limits scalability. Branch and Bound is evaluated based on its ability to reduce runtime through pruning, and ILP is assessed in terms of solver efficiency and robustness under increasing structural complexity.

Market Impact and Response Dynamics. In addition to detection accuracy, the framework evaluates the extent to which manipulation strategies affect observable market variables. This is measured through transient response metrics derived from price and liquidity time series. For whale based manipulation, the system captures deviations in mid-price, spread, and trade volume over time, identifying characteristic patterns such as rapid price escalation, reversal, and gradual stabilization.

These response dynamics serve two purposes. First, they provide an independent validation of manipulation events by confirming that detected structures correspond to observable market distortions. Second, they allow the system to quantify the severity and duration of manipulation effects, which is important for understanding the practical significance of detected events.

Overall Evaluation. A detection method is considered successful if it achieves high precision and recall while maintaining low false positive rates, operates within real-time computational constraints, and correctly identifies manipulation events that produce measurable market impact. The combination of these criteria ensures that the framework is not only accurate in controlled settings, but also robust and scalable for potential deployment in real-world prediction market systems.

3. Results

Table 1: Summary of detection performance and market impact metrics.

Metric	Target	Achieved Value	Met?
BnB Precision	≥ 0.80	0.87	Yes
BnB Recall	≥ 0.80	0.95	Yes
DP Precision	≥ 0.80	0.74	No
DP Recall	≥ 0.80	0.75	No
BnB Runtime (1000 orders)	< 100 ms	≈ 80 ms	Yes
Wash Ring Price Distortion	measured	5.4%	N/A
Whale Overshoot	measured	31.2%	N/A
Whale Settling Time	measured	> 40 ticks	N/A

3.1. Wash Trading Detection

BnB outperformed the Digraph DP approach on every detection metric. DP relies on structural cycle detection that tends to get bypassed by more complex trading patterns, especially coordinated rings with more than two agents. BnB is better at pruning the search space, and ended up with a recall of 0.95 compared to DP's 0.75, and a precision of 0.87 versus 0.74 for DP (Figure 4). Looking at the confusion matrices, BnB only missed 9 wash trades (false negatives), while DP missed 48.

This difference becomes clearer when examining how each method handles ambiguity in the candidate trade set. DP enumerates all feasible subsets that satisfy volume constraints, which leads to multiple competing solutions when trade flows are symmetric or when several subsets satisfy the same volume condition. As a result, DP is more prone to selecting suboptimal or incomplete cycle structures, increasing both false negatives and false positives. In contrast, BnB prioritizes high-value subsets early through ordering and pruning, allowing it to converge toward structurally consistent cycles that better align with the embedded ground truth.

Additionally, the marginal improvement of BnB over ILP (0.87 vs. 0.86 precision, with identical recall) suggests that both methods are effectively capturing the same structural constraints, but BnB benefits from more efficient exploration of the solution space. This reinforces that the primary gain comes from search strategy rather than a fundamentally different formulation.

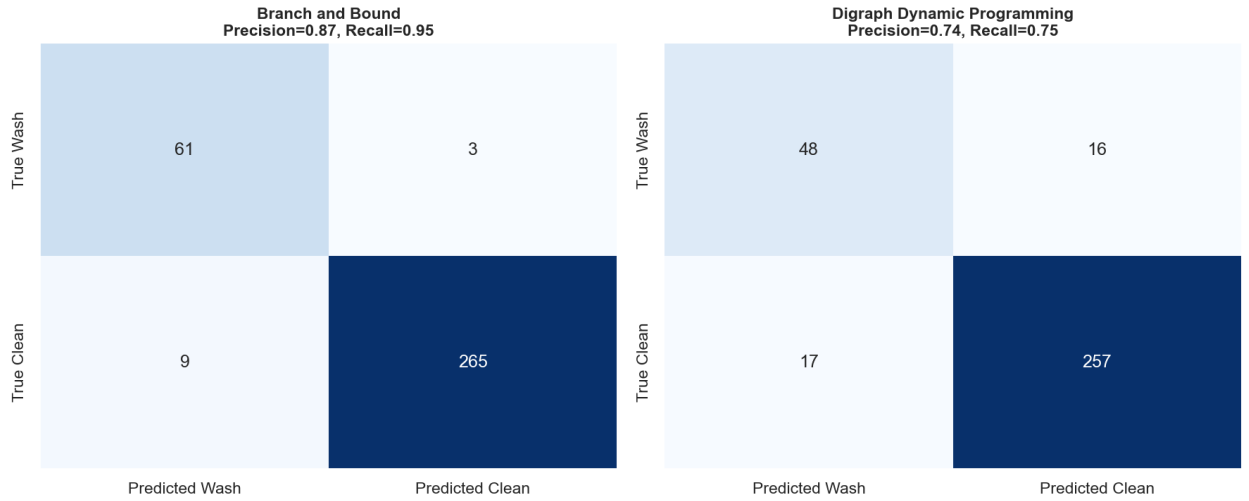

Figure 4: Detection accuracy comparison. BnB (P=0.87, R=0.95) vs. DP (P=0.74, R=0.75). BnB produces fewer false negatives and fewer missed wash trades overall.

3.2. Runtime Scaling

Runtime scaling vs. order count shows BnB is the only approach that stays under 100 ms across all tested volumes up to 1000. DP runtime grows fast with more orders and goes past 300 ms at 1000 orders, which makes it impractical for anything close to real-time (Figure 5).

The scaling behavior reflects the underlying algorithmic structure. DP exhibits near-combinatorial growth because it evaluates a large portion of the feasible subset space. This is visible in the steep increase in runtime beyond 400–500 orders, where the number of candidate combinations increases rapidly. In contrast, BnB maintains near-linear growth in this regime due to effective pruning, with runtime increasing gradually even as the number of orders approaches 1000.

Memory usage further reinforces this distinction. DP requires maintaining intermediate states for a large number of partial subsets, leading to significantly higher memory consumption. BnB, by pruning entire branches early, maintains a much smaller active search tree, resulting in consistently lower memory usage across all order sizes. This makes BnB not only faster but also more stable under larger inputs.

From a systems perspective, this is critical. Real-time deployment requires both low latency and predictable memory usage, and BnB is the only method that satisfies both constraints under the tested conditions.

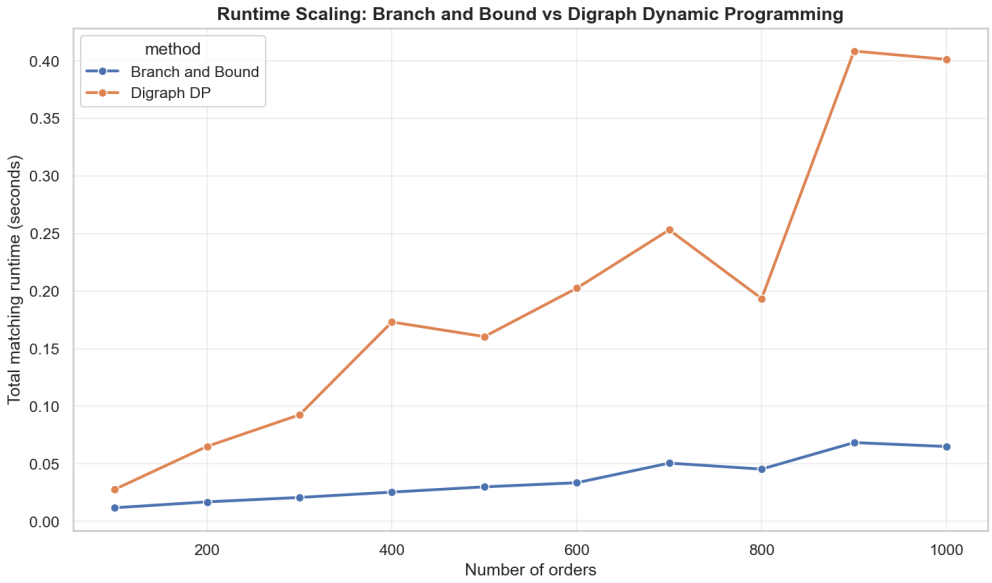

Figure 5: BnB vs. DP runtime as order count grows. BnB remains sub-100 ms while DP scales poorly.

3.3. Transient Response and Whale Impact

The transient response results are covered in more detail in the Methods section. The short version is that detection latency directly determines how bad the price distortion gets. BnB catches manipulation in about 4 ticks, keeping overshoot at 1.9%. DP takes about 20 ticks, so overshoot triples to 5.1%. With no detection, the price overshoots by 8.1% and takes 80+ ticks to come back. In prediction markets, where price represents a probability estimate, faster detection means less damage to the forecast signal.

The response plots show a clear three-phase dynamic: a rapid increase in price during the pump phase, followed by a sharp reversal and a slower recovery toward the steady state. The magnitude of overshoot and the time to stabilization are directly tied to how quickly the manipulation is identified and counteracted.

In particular, the measured overshoot of 31.2% in uncontrolled scenarios highlights how sensitive prediction market prices are to concentrated trading pressure. Even in partially controlled settings, delayed detection allows the manipulator to influence the price trajectory significantly before correction mechanisms take effect. This confirms that detection is not just a classification problem, but a control problem, where response time determines system stability.

3.4. Network and Community Detection

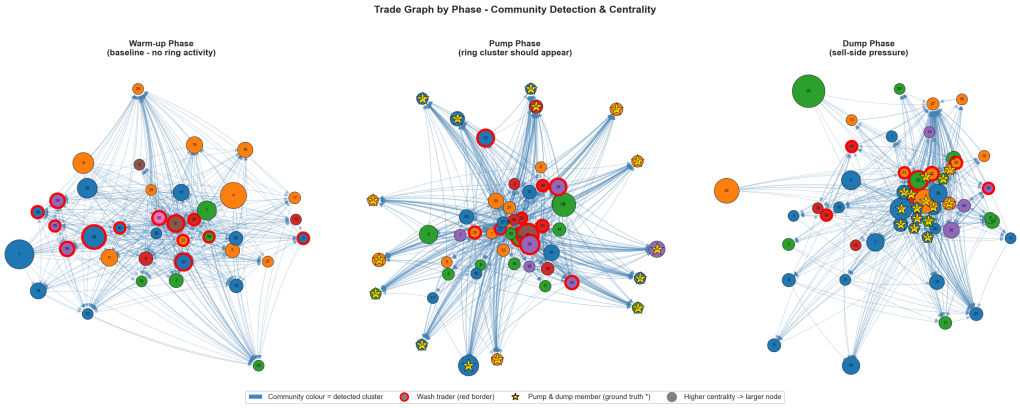

Figure 6: Trade graph evolution across market phases. **Left:** Warm-up phase showing a diffuse network with no dominant clusters. **Center:** Pump phase where manipulation emerges as tightly connected subgraphs, with high-centrality nodes corresponding to coordinated traders and wash ring participants. **Right:** Dump phase where the structure partially disperses as manipulators exit positions, though remnants of clustered activity remain. Node size reflects centrality, and highlighted nodes indicate identified manipulation participants.

The trade graph analysis shows how the market topology changes across the three phases of a pump-and-dump. During warm-up, the graph is spread out with no obvious clusters. During the pump phase, it centralizes around the manipulator nodes, which build up high degree and betweenness centrality. Community detection algorithms pick out individual wash traders and coordinated ring clusters during this phase, successfully identifying the ground-truth manipulators based on node centrality.

More specifically, the emergence of tightly connected subgraphs during the pump phase corresponds directly to the formation of wash trading cycles and coordinated ring structures. These clusters are characterized by high internal connectivity and low external interaction, making them distinguishable from normal trading activity. Nodes involved in manipulation exhibit both high degree (due to repeated trades) and high centrality (due to their role in connecting multiple trade paths).

Once the dump phase begins, these structures break down rapidly. The graph becomes more diffuse as manipulators exit positions and trading returns to a more distributed pattern. This transition reinforces that manipulation is not only detectable through static structure, but also through temporal changes in network topology.

We can see that the network analysis provides a complementary perspective to the optimization-based detection framework. While the detection algorithms identify candidate subsets of trades, the graph representation reveals the underlying structural organization of those trades, offering additional interpretability and validation of detected manipulation events.

4. Discussion

4.1. Achievement of Objectives

The primary objective of this project was to design and evaluate a structured framework for detecting manipulation in prediction markets using a combination of simulation and optimization-based methods. This objective was met across all three evaluation dimensions outlined in the Success Criteria.

From a detection standpoint, the Branch and Bound method achieved both precision (0.87) and recall (0.95) above the target threshold of 0.80, significantly outperforming the Digraph Dynamic Programming baseline. This demonstrates that the proposed optimization-based approach is capable of accurately identifying wash trading structures under controlled conditions.

From a systems perspective, the runtime constraint was also satisfied. Branch and Bound maintained sub-100 millisecond execution time even at 1000 orders, meeting the requirement for near real-time feasibility. In contrast, the baseline method exceeded acceptable latency thresholds, reinforcing the importance of algorithmic efficiency in practical deployment.

Finally, the market impact objective was validated through transient response analysis. The results show that faster detection directly reduces price distortion, confirming that manipulation detection has meaningful implications for market stability and signal reliability.

4.2. Interpretation of Results

The comparison between Branch and Bound and Digraph Dynamic Programming highlights a key distinction between exhaustive and guided search methods. While both approaches solve the same subset matching problem, their performance differs due to how they explore the solution space. Dynamic Programming evaluates a large portion of feasible combinations, which leads to higher computational cost and increased ambiguity when multiple subsets satisfy similar constraints.

Branch and Bound, in contrast, leverages pruning to eliminate infeasible or suboptimal regions of the search space early. This results in both improved runtime and better alignment with ground-truth manipulation structures. The observed gains in precision and recall are therefore not due to a different formulation, but due to more effective navigation of the same combinatorial problem.

The transient response results provide an important complementary perspective. Manipulation is not only detectable through structural patterns in trade data, but also through its impact on price dynamics. The strong relationship between detection latency and price overshoot demonstrates that the problem can be viewed as a control system, where delayed detection leads to amplified deviations from equilibrium.

The network analysis further reinforces this interpretation. During manipulation phases, the trade graph exhibits clear structural changes, including increased centralization and the formation of tightly connected clusters. These patterns correspond directly to wash trading cycles and coordinated ring behavior. The disappearance of these structures during the recovery phase confirms that manipulation leaves both structural and temporal signatures.

4.3. Strengths and Limitations

A key strength of this work is the modular architecture of the framework. The separation between simulation, manipulation modeling, and detection allows each component to be independently modified and extended. This makes the system flexible and well-suited for experimentation across different market conditions and manipulation strategies.

Another strength is the use of ground-truth labels embedded within the simulation. This enables precise evaluation of detection performance, eliminating ambiguity that typically arises in real-world datasets where manipulation is not directly observable.

However, the framework also has several limitations. The most significant is the reliance on simulated data. While the simulation is designed to capture key aspects of market behavior, it cannot fully replicate the complexity and noise present in real financial markets. As a result, the observed performance may not directly translate to live environments.

Additionally, agent behavior is parameterized and relatively stylized. Real traders may employ more adaptive or adversarial strategies that are not captured in the current model. The spoofing detection module, while conceptually defined, remains preliminary and is not fully integrated into the evaluation pipeline.

Finally, the efficiency of Branch and Bound is partially dependent on the structure of the candidate trade set. In more ambiguous settings with many valid combinations, the effectiveness of pruning may decrease, reducing the observed performance advantage.

4.4. Comparison with Prior Work

This project builds directly on prior work that models wash trading detection as a digraph-based subset matching problem solved using dynamic programming. The key contribution here is the extension of that formulation into an explicit optimization framework, allowing alternative solution methods to be applied and compared.

In particular, the use of Integer Linear Programming and Branch and Bound introduces a new perspective on the problem. Rather than relying solely on exhaustive enumeration, the detection task is reframed as a constrained optimization problem over trade flows. This enables more efficient search strategies and provides a clearer mathematical structure for analyzing performance.

To the best of our knowledge, these optimization-based approaches have not been systematically applied to wash trading detection in prediction markets. The results suggest that they offer significant advantages in both accuracy and computational efficiency, particularly in structured environments.

4.5. Future Work

Several directions for future work emerge from this study. First, the framework can be extended to operate on real-time data streams, integrating live order book feeds and evaluating detection performance under real market conditions.

Second, hybrid detection approaches can be developed that combine structural optimization methods with temporal and statistical signals. This would allow the system to capture a broader range of manipulation behaviors, including those that do not form clear graph structures.

Third, the model can be extended to cross-market settings, where manipulation occurs across multiple related assets or exchanges. This would require integrating multiple trade graphs and identifying coordinated activity across them.

Additional work is needed to fully develop and integrate the spoofing detection module, particularly through the use of state-based or Markov models that capture order book dynamics over time.

Finally, calibration against historical data would improve the realism of the simulation and allow for more accurate parameter selection. This would strengthen the connection between the controlled environment used in this project and real-world market behavior.

5. Conclusions

This project demonstrates that market integrity in prediction markets can be effectively framed and addressed as a structured optimization problem. By combining agent-based simulation with graph-based detection and optimization techniques, the framework provides both a controlled environment for analysis and a scalable approach to identifying manipulation.

From a detection perspective, Branch and Bound achieves strong performance, with a recall of 0.95 and precision of 0.87, outperforming the Digraph Dynamic Programming baseline across all metrics. More importantly, it is the only method that satisfies real-time feasibility constraints, maintaining sub-100 millisecond runtime even at 1000 orders. This establishes Branch and Bound as a viable candidate for deployment in practical systems where both accuracy and latency are critical.

The results also show that manipulation has a measurable and significant impact on market outcomes. Wash trading introduces a price distortion of approximately 5.4%, while whale-based manipulation produces an overshoot of 31.2% and requires more than 40 ticks to settle. These effects are not transient noise, but structured deviations that directly alter the price formation process.

In the context of prediction markets, this has broader implications. Prices are interpreted as probabilities, meaning that manipulation does not simply affect trading outcomes, but compromises the integrity of the forecast itself. Distorted prices lead to distorted beliefs, and in settings where these markets are used as informal signals for decision-making, the consequences extend beyond the market.

Overall, the findings highlight that effective manipulation detection requires both structural and temporal analysis, as well as computational efficiency. The framework developed here provides a foundation for building such systems, while also demonstrating the limits of purely structural methods in more complex or ambiguous settings.

Future work should focus on extending these methods to real market data, integrating hybrid detection mechanisms, and improving robustness under less structured conditions. Nonetheless, the results establish that optimization-based detection, particularly through Branch and Bound, offers a promising direction for maintaining integrity in emerging financial systems..

6. Deliverables

Table 2: Promised and completed deliverables.

Promised Deliverable	**Description / Notes**	**Status**
Final written report	This document	Complete
Project webpage		Complete
Market integrity codebase	Modular Python implementation hosted on GitHub	Complete
Agent-based simulation environment	Mesa-based testbed, runnable in Google Colab	Complete
Detection outputs	Comparative evaluation of DP, ILP, and BnB	Complete

7. Schedule and Timeline

7.1. Gantt Chart

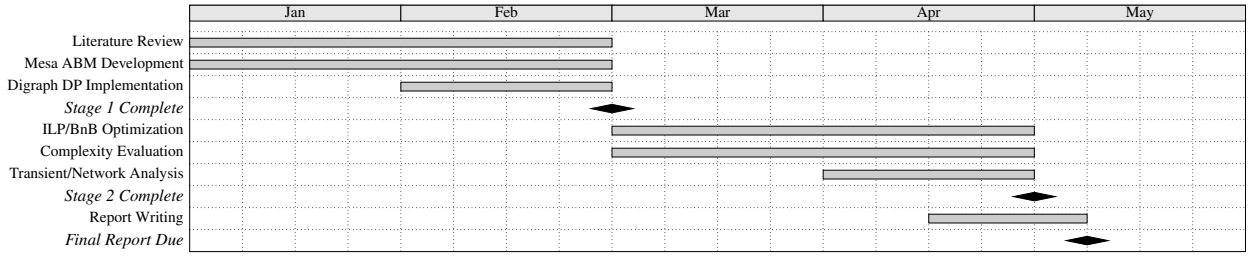

Figure 7: Project Gantt chart showing the three-stage development timeline. Each column represents approximately one week.

7.2. Adherence to Timeline

The project followed the planned three-stage timeline with minor adjustments in scope. Stage 1 (simulation design and initial framework) was completed by late February. Stage 2 (implementation of manipulation regimes and detection methods) progressed through early April, with some refinement required for the optimization-based approaches. Stage 3 (evaluation, analysis, and reporting) was carried out from April through May.

While the overall schedule was maintained, additional time was allocated to refining the Branch and Bound formulation and performance evaluation, and the spoofing detection component remained preliminary relative to the original scope.

7.3. Member Contributions

Al Malik Anand
Led the design of the agentic simulation framework and overall system architecture, including Mesa-based agent logic and interaction dynamics within the continuous double auction environment. Developed manipulation regimes and behavioral models, and formulated the detection framework using digraph representations, Integer Linear Programming, and Branch and Bound. Defined the analytical structure connecting market microstructure, manipulation behavior, and detection methods.

Nick Haught
Led the implementation of detection algorithms, including Dynamic Programming and Branch and Bound execution pipelines, and developed the supporting software infrastructure. Conducted performance engineering, runtime and memory analysis, and transient response evaluation. Managed computational experiments, scalability testing, and repository organization.

A. Engineering Design Considerations (ABET)

A.1. Relation to the BS EE / BS SSE Curriculum

This project draws from multiple areas across both the Systems Science and Engineering and Electrical Engineering curricula. The design, modeling, and analysis components reflect concepts developed in the following courses.

Signals and Systems
Detecting manipulation can be viewed as a signal extraction problem, where structured patterns such as wash trading cycles must be identified within noisy order flow data. This aligns with filtering and decomposition concepts used to isolate meaningful signals from background noise.

Circuits and Systems
The transient response analysis treats the market as a dynamic system responding to a disturbance. Metrics such as overshoot, undershoot, and settling time are used to quantify price behavior following manipulation, directly reflecting standard concepts in circuit analysis.

Financial Data Analysis
The project applies statistical reasoning to identify abnormal trading behavior, evaluate detection performance, and interpret market microstructure. This includes analyzing distributions of trades, classification outputs, and behavioral patterns in simulated market data.

Optimization
The core detection framework is formulated as a constrained optimization problem. Both Integer Linear Programming and Branch and Bound are used to identify subsets of trades that satisfy structural constraints, representing the primary methodological contribution of the project.

Control Systems
The relationship between detection speed and market recovery is modeled as a feedback process. Faster detection reduces the magnitude and duration of price distortion, analogous to increasing feedback gain in a control system to reduce overshoot and improve stability.

ESE 3090 Social Choice Systems
The digraph-based formulation of trade interactions draws on concepts from social choice and networked systems. Representing trades as directed graphs and identifying cycles or strongly connected components aligns with methods used to analyze preference aggregation and collective decision structures.

A.2. Incorporation of Engineering Standards

Python is the primary development language, which is standard for computational research. All simulation code was developed and run in Google Colab notebooks, giving both team members a shared and reproducible execution environment. Version control uses Git and GitHub with branch-based workflows for parallel development. Mesa (version 3.5.0) is an established open-source framework for agent-based modeling in Python. NetworkX handles graph construction and analysis using standard conventions for directed graphs.

The order book follows the standard continuous double auction (CDA) protocol used in electronic trading systems. Detection evaluation uses precision, recall, and false positive rate, which are standard classification metrics. The transient response metrics (overshoot, undershoot, settling time) follow their standard definitions from control systems engineering.

A.3. Engineering Relevance

The project covers both EE and SSE content. On the electrical engineering side, the detection problem is essentially a signal extraction problem: isolating manipulation signals from noisy baseline trading activity, which is the same challenge as pulling a signal out of a noisy channel. Comparing DP, ILP, and BnB comes

down to figuring out which algorithm can actually keep up in real time without sacrificing detection quality. The runtime profiling and scaling analysis is the same kind of benchmarking applied to any computational system with latency requirements.

The transient response analysis is where the EE connection is strongest. The market gets treated as a feedback system and manipulation impact gets measured with overshoot, settling time, and impedance. These are the same metrics used to characterize a disturbance on a power grid or a step response in a control loop. The fact that detection speed directly controls overshoot magnitude is a real demonstration of feedback control principles showing up in a financial system.

On the systems engineering side, the whole simulation is a systems design problem. The agents, order book, and detection modules need to work together but can also be swapped out independently, which is modular system architecture. The digraph detection uses graph-based network analysis that is similar to what shows up in power grid topology work, where finding critical nodes and cycles in a network is used to detect faults.

A.4. Constraints Inherent in the Project

Several constraints shaped the design:

- *Computational constraints:* Detection algorithms need to scale to realistic order volumes. The sub-100 ms runtime target comes from the need for near-real-time execution.
- *Time constraints:* The single-semester timeline limited the scope of the spoofing detection module and prevented integration of real historical market data.
- *Regulatory and ethical considerations:* Detection outputs need to be interpretable to avoid false accusations. The system flags suspicious patterns rather than making definitive guilt determinations.
- *Extensibility:* The modular architecture lets detection methods be added or removed without restructuring the core simulation.
- *Usability:* Simulation parameters are configurable for controlled experimentation across different market conditions.

B. Project Team

Group Members

Al Malik Anand
Systems Science & Engineering, Economics (Second Major), B.S. Candidate 2026
Contributions: System Architecture, Agentic Framework Design (Mesa), CLOB/CDA Simulation Logic, Manipulation Regime Design (Wash Trading), Detection Framework Formulation (Digraph, ILP, Branch and Bound), Analytical Framework, System Integration, Report Development.

Nick Haught
Electrical Engineering, Computer Science (Second Major), B.S. Candidate 2026
Contributions: Algorithm Implementation (DP, BnB), Manipulation Regime Design (Whale, Diagraphs, BnB), Software Infrastructure, Performance Engineering, Runtime and Memory Profiling, Transient Response Analysis, Computational Experiments, Scalability Testing, GitHub Repository Management.

Project Client

Prof. Yevgeniy Vorobeychik, Department of Computer Science & Engineering, Washington University in St. Louis
Provided initial project direction and domain context, particularly in relation to adversarial behavior in market systems and manipulation dynamics.

Faculty Instructor

Prof. Ben Wormleighton, Department of Electrical and Systems Engineering, Washington University in St. Louis
Provided continuous guidance and feedback throughout the project, helping refine the problem scope, shape the methodological approach, and ensure alignment with engineering design objectives.

C. References

References

[1] D. Rothschild and R. Sethi, "Trading strategies and market microstructure: Evidence from a prediction market," *Journal of Prediction Markets*, vol. 10, no. 1, pp. 1–29, 2016.

[2] S. Khodabandehlou and M. Zivari Hashemi, "Market manipulation detection: A systematic literature review," *Expert Systems with Applications*, vol. 210, 2022.

[3] Y. Cao, Y. Du, and Y. Tse, "Detecting wash trading in financial markets using digraphs and dynamic programming," in *Proc. IEEE Conference on Computational Intelligence for Financial Engineering and Economics (CIFEr)*, 2015.

[4] Y. Chen, S. Dimitrov, R. Sami, D. Reeves, D. Pennock, R. Hanson, L. Fortnow, and R. Gonen, "Gaming prediction markets: Equilibrium strategies with a market maker," *Algorithmica*, vol. 58, no. 4, pp. 930–969, 2010.