Stock Market Tangle: A Graph-Theoretic Model of Financial Markets

Algorithm Analysis and Design

Team: Queue-ties

Members: Avani Sood, Guntesh Singh, Khushi Dhingra, Saanvi Jain, Vaibhavi Kolipaka

Abstract

Financial markets exhibit dense correlation structures that traditional variance-covariance models often obscure. This study models inter-stock dependencies as weighted, undirected graphs to analyze market topology across four distinct regimes: Stable, Normal, Volatile, and Crash. We implemented seven fundamental algorithms—Union-Find, Breadth-First Search (BFS), Depth-First Search (DFS), PageRank, Girvan-Newman, Louvain, and Node2Vec—from first principles. The evaluation framework balances computational efficiency, structural fidelity, and financial relevance. Empirical benchmarks reveal critical trade-offs: Union-Find delivers sub-millisecond component detection (<1 ms), whereas Girvan-Newman's $O(V^2E)$ complexity renders it prohibitive (32.97 s) on dense, 500-stock graphs. Conversely, Louvain optimization achieves scalable community detection, though it faces resolution limits in highly interconnected regimes. Connectivity analysis demonstrates that stable markets coalesce into a single component with complete integration, while crash scenarios fragment into isolated vertices. By integrating these metrics, we propose a graph-theoretic analytics pipeline that identifies bridge stocks, quantifies sector cohesion, and generates diversification-aware portfolio recommendations that improve simulated Sharpe ratios by 0.18 under stress conditions.

1. Introduction

On March 12, 2020, global equity markets experienced their sharpest single-day decline since 1987. Correlations across seemingly unrelated sectors spiked toward unity, causing traditional diversification strategies—which assume independent asset behaviors—to fail precisely when needed most. In reality, financial markets function as dense, dynamic networks where shocks propagate through complex pathways that static factor models cannot capture.

This project reframes stock market analysis through the lens of **graph theory**, representing securities as nodes and pairwise correlations as weighted edges. Unlike "black-box" machine learning models, graph algorithms provide interpretable insights into systemic risk, contagion pathways, and portfolio fragility.

The investigation addresses three core objectives:

1. **Computational Efficiency:** Characterizing algorithmic scalability across graph sizes (100–500 nodes) to distinguish between real-time monitoring tools and batch-mode analytics.
2. **Structural Accuracy:** Validating that detected communities reflect genuine market organization using modularity scores (Q) and Normalized Variation of Information (NVI).
3. **Financial Relevance:** Translating abstract structural insights into actionable intelligence, such as bridge-stock alerts and diversification-aware recommendations.

The algorithmic suite spans five functional categories. Union-Find, BFS, and DFS form the connectivity foundation, answering questions about market fragmentation and integration with minimal computational overhead. PageRank identifies systemically important stocks whose movements influence broad market segments. Girvan-Newman and Louvain detect community structures, revealing sectoral boundaries and overlapping risk clusters. Node2Vec learns distributed representations that encode both local neighborhoods and global market positions, enabling similarity-based portfolio construction.

The following sections detail our algorithmic design, implementation challenges, and empirical findings, concluding with a unified framework for deploying these tools in real-world portfolio management.

2. Theoretical Framework and Algorithms

We selected a suite of algorithms spanning connectivity, centrality, and representation learning.

2.1 Union-Find with Path Compression

The Union-Find data structure, also known as Disjoint-Set Union (DSU), efficiently tracks and merges disjoint sets through two fundamental operations: Find(x) determines which set contains element x by returning its representative root, while Union(x, y) merges the sets containing x and y. In the context of stock market graphs, Union-Find identifies connected components within correlation networks, revealing patterns of market segmentation and integration.

Optimization Techniques. Path compression ensures that during Find operations, all nodes along the search path are directly connected to the root, flattening the tree structure. Subsequent Find queries on the same path then execute in near-constant time. Union-by-rank prevents tree degeneration by attaching the shallower tree under the root of the deeper tree during merge operations.

Complexity Analysis. With both optimizations, each operation achieves $O(\alpha(n))$ amortized time complexity, where $\alpha(n)$ denotes the inverse Ackermann function. This function grows extraordinarily slowly—$\alpha(n) < 5$ for all practical input sizes ($n < 2^{65536}$)—making Union-Find effectively constant-time in practice. Space complexity remains $O(n)$ for the parent and rank arrays.

Financial Application. When edges represent correlations exceeding volatility-adjusted thresholds, each connected component corresponds to a market segment of stocks exhibiting coordinated movements. Component size distributions indicate market concentration: a single dominant component signals high integration, while numerous small components reveal sector fragmentation. Bridge stocks—those whose addition merges previously separate components—represent critical contagion pathways. Tracking component counts over time quantifies market stability: sudden fragmentation during volatile periods indicates correlation breakdown and diversification failure.

2.2 Breadth-First Search

Breadth-First Search explores graphs level by level, visiting all nodes at distance d before advancing to distance $d+1$. This systematic traversal pattern makes BFS particularly suited for computing shortest paths in unweighted graphs and, by extension, minimum-correlation-hop routes between stocks in financial networks.

Algorithm Overview. Starting from a source node, BFS maintains a queue of nodes to visit, marking each as explored before enqueuing its neighbors. The queue ensures that nodes are processed in order of increasing distance from the source. Parent pointers maintained during traversal enable path reconstruction.

Complexity Analysis. BFS visits each vertex exactly once and examines each edge at most twice (once from each endpoint), yielding $O(V + E)$ time complexity. Space complexity is $O(V)$ for the visited set and queue.

Financial Application. BFS on correlation graphs enables several portfolio construction strategies. Shortest correlation paths reveal minimal chains connecting two stocks—for instance, AAPL $\to$ MSFT might route through two intermediate technology equities, indicating indirect exposure. By filtering edges below volatility thresholds, BFS identifies low-volatility diversification routes that remain stable during market stress. Reachability analysis determines whether two stocks are connected through any correlation chain, answering questions about portfolio independence. Distance-based diversification exploits the principle that stocks separated by greater BFS distances exhibit weaker correlations, making them natural hedging pairs. Average path length across the entire graph quantifies overall market integration: shorter paths indicate tighter coupling and faster shock propagation.

2.3 Depth-First Search

Depth-First Search explores graphs by recursively following each branch to its maximum depth before backtracking. This exploration pattern proves particularly effective for cycle detection and component enumeration in financial correlation networks.

Algorithm Overview. Starting from a source node, DFS marks the current node as visited and recursively explores each unvisited neighbor. When no unvisited neighbors remain, the algorithm backtracks to the most recent node with unexplored branches. The recursion stack implicitly maintains the current exploration path.

Complexity Analysis. DFS visits each vertex once and traverses each edge at most twice, achieving $O(V + E)$ time complexity. Space complexity is $O(V)$ for the visited set and recursion stack, with stack depth proportional to the longest path in worst-case scenarios.

Financial Application. DFS provides complementary connectivity analysis to BFS. Cycle detection identifies circular correlation dependencies (A$\to$B$\to$C$\to$A) that amplify risk during market stress—when one stock in a cycle declines, correlated movements propagate around the loop, potentially triggering feedback effects. Component enumeration via repeated DFS from unvisited nodes efficiently partitions the graph into isolated sectors. The connectivity ratio—defined as the number of edges in the largest component divided by the maximum possible edges—measures market cohesion. During stable periods, this ratio approaches unity as correlations span most stock pairs. During crashes, the ratio collapses toward zero as correlations dissolve and the market fragments. Path enumeration capabilities support stress testing by revealing all possible correlation routes through which shocks might propagate between specific stocks.

2.4 PageRank

PageRank, originally developed by Brin and Page (1998) for web search, adapts naturally to financial networks by treating correlation strength as analogous to hyperlink authority. The algorithm identifies systemically important stocks whose price movements disproportionately influence broader market dynamics.

Algorithm Overview. PageRank models a random walker traversing the correlation graph with probability $d$ (the damping factor, typically 0.85) or teleporting to a uniformly random node with probability $1-d$. Initially, all nodes receive equal rank $1/N$. Iteratively, each node's rank is updated as the teleportation baseline $(1-d)/N$ plus the damping factor times the sum of incoming ranks weighted by neighbor degree: $rank(v) = (1-d)/N + d \times \Sigma(rank(u)/degree(u))$ for all edges $u \to v$. Iteration continues until rank changes fall below a convergence threshold $\epsilon$.

Complexity Analysis. Each iteration requires $O(V + E)$ operations to update all ranks. Convergence typically occurs within $k = 20–50$ iterations depending on graph density and damping factor, yielding $O(k(V + E))$ total complexity. Space complexity is $O(V)$ for current and previous rank vectors.

Financial Application. High-PageRank stocks occupy central positions in the correlation network, meaning their price movements propagate influence through many paths to many other securities. These stocks represent contagion hubs: negative shocks to high-PageRank equities trigger cascading effects throughout the market. Portfolio managers overweighting high-PageRank stocks inadvertently concentrate systemic risk, as their holdings become vulnerable to broad market downturns. Conversely, ranking distributions provide early warning signals—dramatic rank shifts between stable and volatile periods indicate structural changes in market organization. The weighted variant employed here uses correlation strength rather than binary connectivity, ensuring that strong correlations contribute more influence than weak ones.

2.5 Girvan-Newman Community Detection

The Girvan-Newman algorithm detects communities by iteratively removing edges with the highest betweenness centrality—defined as the fraction of shortest paths passing through each edge. This divisive hierarchical approach naturally reveals "bridge" edges connecting dense clusters, making it particularly valuable for identifying inter-sector stocks in financial networks.

Algorithm Overview. At each iteration, edge betweenness centrality is computed for all remaining edges using Brandes' fast algorithm (2001), which employs BFS from every node to accumulate shortest-path counts. The edge with maximum betweenness is removed, and the process repeats until the graph fragments into isolated nodes or modularity begins decreasing. Modularity—the fraction of edges within communities minus the expected fraction under random wiring—serves as the quality metric for selecting the optimal partition from the dendrogram of successive divisions.

Complexity Analysis. Brandes' algorithm computes all-pairs betweenness in $O(VE)$ time. Since Girvan-Newman may remove up to E edges, worst-case complexity is $O(VE^2)$, though early termination based on modularity decline typically reduces practical runtime. Space complexity is $O(V + E)$ for the adjacency list and betweenness accumulator.

Financial Application. Bridge edges in correlation graphs represent stocks with exposure to multiple sectors—for instance, Apple (AAPL) might bridge technology and consumer discretionary sectors, while JPMorgan Chase (JPM) connects financials and real estate. These bridge stocks become critical during contagion events, as shocks in one sector propagate through bridges to previously uncorrelated sectors. Removing high-betweenness edges during market stress tests reveals which connections would most effectively contain systemic risk. The hierarchical dendrogram produced by Girvan-Newman also provides a natural visualization of sector taxonomy at multiple granularity levels, from broad industry groups down to narrow subsectors.

2.6 Louvain Modularity Optimization

The Louvain algorithm maximizes modularity through a two-phase iterative process: local optimization, where individual nodes migrate to neighboring communities that maximize modularity gain, and aggregation, where detected communities collapse into super-nodes and the process repeats on the compressed graph. This greedy heuristic achieves near-linear runtime scaling on large networks while typically matching or exceeding the quality of divisive methods like Girvan-Newman.

Algorithm Overview. Phase 1 iteratively evaluates modularity gain ΔQ for moving each node to each neighboring community, selecting the move with maximum positive gain. This process continues until no node can improve modularity by changing communities. Phase 2 then aggregates nodes within each community into a single super-node, with edge weights between super-nodes equal to the sum of inter-community edge weights from Phase 1. The algorithm recurses on this coarse-grained graph until modularity converges, typically within 3–5 iterations.

Complexity Analysis. Each local move requires evaluating $O(degree)$ potential community assignments. In practice, runtime scales near-linearly with the number of edges for sparse graphs and remains efficient even on dense networks due to rapid convergence. Theoretical guarantees are heuristic, but empirical studies confirm sub-quadratic scaling on most real-world networks.

Financial Application. Louvain identifies "market tribes"—clusters of stocks exhibiting strong internal correlations and weak external correlations. These tribes often correspond to sectors (technology, healthcare, energy) or behavioral groups (growth stocks, value stocks, defensive equities). Unlike rigid sector classifications from standard taxonomies, Louvain-detected communities reflect actual correlation patterns that evolve dynamically with market conditions. High modularity scores indicate well-defined sector boundaries and effective diversification opportunities, while low modularity during crashes signals correlation convergence and diversification breakdown.

2.7 Node2Vec Embeddings

Node2Vec, introduced by Grover and Leskovec (2016), learns continuous vector representations of nodes by optimizing a Skip-gram objective on sequences generated through biased random walks. The resulting embeddings encode both local neighborhood structure and global network position, enabling similarity-based queries and machine learning applications.

Algorithm Overview. Biased random walks are controlled by two parameters: the return parameter $p$ governs the probability of returning to the previous node (lower $p$ encourages breadth-first exploration), while the in-out parameter $q$ governs the probability of exploring distant nodes versus staying near the starting point (lower $q$ encourages depth-first exploration). For each node, multiple walks of fixed length are generated, producing sequences analogous to sentences in natural language processing. These sequences then train a Skip-gram model with negative sampling, where the objective is to predict context nodes (neighbors in the walk) from target nodes. Stochastic gradient descent updates two embedding matrices—input embeddings $W_{in}$ and context embeddings $W_{out}$—until convergence.

Complexity Analysis. Generating walks requires $O(walk\_length \times walks\_per\_node \times V)$ time. Training the Skip-gram model takes $O(walk\_length \times walks\_per\_node \times V \times embedding\_dimension \times negative\_samples)$ time. Space complexity is $O(V \times embedding\_dimension)$ for the learned representations.

Financial Application. Node2Vec embeddings facilitate similarity-based portfolio construction. Stocks with similar embeddings—measured via cosine similarity—tend to occupy similar structural positions in the correlation network and exhibit comparable behavioral patterns. This enables several applications: identifying substitute holdings when rebalancing portfolios, discovering diversification candidates by selecting stocks with dissimilar embeddings, and clustering stocks into behavioral groups that transcend traditional sector classifications. Post-filtering based on mean-variance optimization refines raw embedding similarity into risk-adjusted portfolio recommendations. Preliminary results indicate 4× higher hit rates compared to random baselines, with simulated Sharpe ratio improvements of 0.18.

3. Methodology and Implementation

3.1 Data Generation and Graph Construction

We generated synthetic return series using a multifactor model calibrated to S&P 500 empirical data (2015–2020). The covariance matrix Σ was converted to a correlation matrix P. The adjacency matrix A was constructed by thresholding:

$$A_{ij} = \begin{cases} |\rho_{ij}| & \text{if } |\rho_{ij}| > \tau \\ 0 & \text{otherwise} \end{cases}$$

Where τ varies by market regime.

A unified Graph class (defined in src/graph.py) serves as the substrate for all algorithms, maintaining symmetric adjacency lists with floating-point weights.

Key Engineering Challenges:

- **From-Scratch Philosophy:** All algorithms were implemented manually without external graph libraries (e.g., NetworkX) to ensure O-notation transparency.
- **Girvan-Newman Optimization:** We implemented Brandes' algorithm for betweenness centrality. On dense graphs, we optimized path counting to prevent integer overflow.
- **Numerical Stability:** In PageRank, damping factors close to 1.0 caused rank drift; we standardized $d=0.85$ to balance convergence speed with accuracy.
- **Louvain Resolution:** Initial implementations yielded $Q=0$ on dense graphs due to the "resolution limit" of modularity. We introduced a resolution parameter γ to sensitize the algorithm to smaller communities in highly connected stable markets.

4. Experimental Setup

Synthetic Data Generation:

We utilized a multifactor model to generate stock returns, calibrated to empirical correlations observed in S&P 500 data (2015–2020). Edge weights represent the absolute value of Pearson correlations.

Market Scenarios & Topology:

- **Stable:** Threshold $\rho > 0.35$. High integration (Avg Degree ≈ 9). Single giant component.
- **Normal:** Threshold $\rho > 0.45$. Sector formation (Avg Degree ≈ 4). ~45 components.

- **Volatile:** Threshold $\rho > 0.40$. Correlation breakdown (Avg Degree < 1). High fragmentation.
- **Crash:** Threshold $\rho > 0.30$. Complete disconnect (0 edges). Diversification failure.

Evaluation Metrics:

- **Efficiency:** Wall-clock runtime and Peak RSS memory.
- **Accuracy:** Modularity (Q) for community quality; Normalized Variation of Information (NVI) for partition stability.
- **Financial:** Recommendation Hit Rate and simulated Sharpe Ratios (via Monte Carlo backtesting).

---

5. Results and Analysis

5.1 Computational Efficiency

Table 1 presents the runtime performance on 500-node graphs under the Stable regime (highest edge density).

Table 1: Algorithm Runtime Comparison ($N=500$, Stable)

Algorithm	Runtime	Space Complexity	Scalability Assessment
Union-Find	3.00 ms	$O(V)$	Excellent (Real-time)
BFS / DFS	7.00 ms	$O(V)$	Excellent (Real-time)
PageRank	243 ms	$O(V+E)$	Good (Intraday)
Girvan-Newman	32.97 s	$O(VE^2)$	Poor (Research only)
Louvain	72.26 s	$O(V+E)$	Moderate (Batch)

Node2Vec	72.10 s	$O(V \cdot d)$	Moderate (Batch)

Analysis: The $O(VE^2)$ complexity of Girvan-Newman makes it prohibitive for dense graphs, validating its use primarily for sparse, specific sub-graph analysis. Louvain, while typically fast, exhibited overhead on dense graphs due to the "resolution limit"—the high connectivity required excessive aggregation steps to distinguish communities. Union-Find demonstrated effectively constant time, confirming its utility for high-frequency market monitoring.

5.2 Market Fragmentation and Topology

Connectivity metrics quantify the structural transition of markets during stress events.

- **Stable Markets:** The graph forms a single connected component with an average shortest path of **1.84 hops**. This "small-world" property implies that shocks propagate globally with minimal attenuation.
- **Normal Markets:** The graph fragments into 45 components with path lengths increasing to **4.25 hops**. These longer paths act as firebreaks, allowing for effective sector-based diversification.
- **Crash Scenarios:** Connectivity collapses to zero (100 isolated components). This validates the financial observation that during liquidity crises, "all correlations go to one" (directionally), but the *structural* correlations used for hedging dissolve.

5.3 Structural Accuracy Metrics

To validate community detection, we compared Girvan-Newman (GN) and Louvain.

- **Modularity (Q):** On sparse graphs (Normal/Volatile), both algorithms achieved high modularity ($Q \in [0.51, 0.81]$). However, on dense stable graphs, GN produced negative modularity scores ($Q \approx -3.5$), indicating over-partitioning. Louvain initially struggled with the resolution limit ($Q \to 0$), but after parameter tuning, it achieved stable partitions ($Q \approx 0.45$).
- **Partition Similarity (NVI):** The Normalized Variation of Information between GN and Louvain partitions was **0.12** (where 0 indicates identical partitions). This low NVI suggests that despite algorithmic differences, both methods converge on similar fundamental market structures, validating the existence of robust "market tribes."
- **PageRank Convergence:** Influence diffusion speeds up with density. Stable graphs converged in 13 iterations, while fragmented normal graphs required 43–67 iterations, highlighting how market integration accelerates information flow.

5.4 Financial Interpretation

The structural metrics translate directly into portfolio management strategies:

1. **Bridge Stock Identification:** Girvan-Newman explicitly identifies edges with high betweenness. In our simulations, removing these "bridge" stocks (e.g., conglomerates) from a portfolio reduced simulated volatility by **15–22%** during sector-specific downturns by severing contagion pathways.
2. **Systemic Risk Scoring:** PageRank scores correlated strongly with empirical volatility. Stocks with scores >0.01 (in 500-node graphs) consistently acted as lead indicators for broad market movements.
3. **Resilient Allocation:** Node2Vec embeddings allowed us to construct portfolios based on "structural dissimilarity." Backtesting these portfolios against random selections yielded a **0.18 increase in Sharpe Ratio** during crash scenarios, proving that graph-based features capture risk dimensions that simple covariance ignores.

---

6. Discussion: Algorithmic Trade-offs

No single algorithm satisfies all financial analytics requirements. The choice depends on the specific risk management objective.

Table 2: Unified Trade-off Framework

Algorithm	Primary Strength	Critical Weakness	Best Use Case
Union-Find	Speed (<1 ms)	Binary output (connected/not)	**Kill-switch**: Detecting sudden market fracture.
PageRank	Global Influence	Sensitive to damping factor	**Risk Weighting**: Adjusting position sizes by systemic risk.
Girvan-Newman	Interpretability (Bridges)	Scalability ($O(VE^2)$)	**Stress Testing**: Analyzing specific contagion paths.

Louvain	Scalability (Large N)	Resolution limit on dense graphs	**Asset Allocation**: Defining dynamic sectors.
Node2Vec	Feature Richness	High training cost	**Stock Selection**: Finding non-obvious diversifiers.

Limitations:

Our study relies on synthetic data assuming Gaussian returns. Real markets exhibit "fat tails" (kurtosis), which may generate stronger transient correlations than our model predicts. Furthermore, the discrete "regime" buckets (Stable vs. Crash) simplify the continuous evolution of real markets.

---

7. Conclusion

This project demonstrates that graph theory offers a potent alternative to traditional financial modeling. By mapping the "Stock Market Tangle," we revealed that market stability is structurally distinct from market stress—not just in volatility, but in topology. Stable markets are "small worlds" where diversification is costly; crash markets are "fragmented archipelagos" where correlations vanish.

Returning to the introduction: Had these tools been active on **March 12, 2020**, the **Union-Find** monitors would have flagged the sudden fragmentation of the correlation graph in milliseconds, while **PageRank** would have identified the specific liquidity hubs driving the contagion.

While **Union-Find** and **DFS** provide the speed necessary for high-frequency alerts, **Louvain** and **Node2Vec** offer the depth required for strategic rebalancing. We conclude that a hybrid pipeline—using lightweight connectivity algorithms for monitoring and heavy-weight community detection for nightly rebalancing—provides the optimal balance of speed and insight for modern portfolio management.

Future Work

Future iterations will incorporate **Dynamic Graph Neural Networks (DGNNs)** to handle temporal evolution without re-computing from scratch, and validatation against high-frequency tick data to capture non-Gaussian tail dependencies.

References

1. Brandes, U. (2001). A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*.
2. Girvan, M., & Newman, M. E. J. (2002). Community structure in social and biological networks. *PNAS*.
3. Grover, A., & Leskovec, J. (2016). node2vec: Scalable feature learning for networks. *KDD*.
4. Blondel, V. D., et al. (2008). Fast unfolding of communities in large networks. *J. Stat. Mech*.