Network Science Applications for E-Commerce Optimization

PPT Link:
https://www.canva.com/design/DAGmmbJBCdA/Fwb5vh8AdqBFEwKsRLaLuw/edit?utm_content=DAGmmbJBCdA&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

1. Introduction

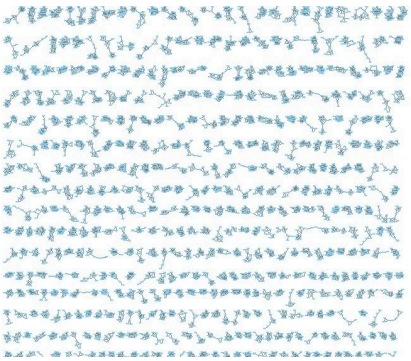
E-commerce platforms increasingly rely on data-driven intelligence to enhance recommendations, manage prices, and ensure fault tolerance. At the heart of this intelligence lies the structure of customer behavior—what products they buy together, how they respond to price changes, and how recommendation engines perform under stress.

This project leverages network science to deliver a set of strategic tools using the Amazon Co-Purchasing Network. Four deliverables are presented:

1. Community-Based Product Bundle Generator
2. Sensitivity Matrix via Influence Diffusion
3. Fault-Tolerant Recommender Design
4. Centrality-Based Influence Simulation

Each component addresses a distinct challenge in e-commerce and is underpinned by scalable graph-theoretic techniques.

2. Dataset Overview



2.1 Main Dataset

- **File**: Database.txt
- **Source**: Stanford SNAP: Amazon0601
- **Structure**:
    - Nodes: Products
    - Edges: “Customers who bought this item also bought” relationships

Metric	Value
Nodes	403,394

Edges	3,387,388

This full dataset was used in influence modeling and robustness analysis.

2.2 Reduced Datasets for Development and Analysis

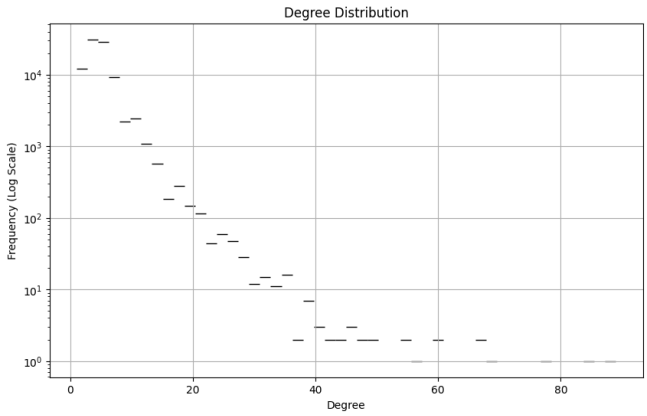
To improve runtime efficiency during experiments and testing, two reduced datasets were created using **degree-based filtering**:

Dataset	Nodes
reduceddataset.txt	~90,000
reduceddataset14k.txt	14,000

The filtering ensured retention of high-connectivity products while reducing computational overhead.

3. Full Graph Analysis & Structural Properties

We conducted an in-depth structural analysis of `reduceddataset.txt` to better understand the underlying properties of the product co-purchasing network.



3.1 Basic Properties

Property	
Number of Nodes	88,687
Number of Edges	221,086
Connectedness	(Disconnected)
Number of Connected Components	2,484
Largest Component Size	72,605 nodes
Average Node Degree	4.99
Graph Density	0.000056

Average Clustering Coefficient	0.5461
Highest Degree Node	Node 458358 (Degree: 89)

3.2 Degree Distribution

- The graph shows a heavy-tailed degree distribution, indicating that a few products are frequently co-purchased with many others.

3.3 Scale-Free Network Check

Using **power-law fitting**:

- Power-law exponent α estimated with x min threshold.
- Statistical comparison with the exponential distribution yielded:
    - $D>0$ and $p<0.05$ → strong evidence for scale-free structure.

The product co-purchasing network is likely scale-free, dominated by hubs.

3.4 Small-World Property Check

As the graph is not fully connected, the standard small-world property test was skipped. In future work, the largest connected component can be isolated and tested using:

- Clustering coefficient CCC
- Average path length LLL
- Comparison to a random graph of same size

4. Deliverable 1: Community-Based Product Bundle Generator

Objective

Automatically identify product bundles based on natural community structures in co-purchase behavior.

Network Science Concept: *Louvain Community Detection*

By applying the Louvain algorithm with a resolution < 1.0, large clusters of frequently co-purchased items were discovered. Each cluster serves as a potential product bundle.

Process Summary

- Constructed product graph from reduceddataset.txt
- Detected communities via Louvain algorithm
- Filtered out small clusters (fewer than 3 nodes)
- Ranked and displayed largest clusters

Example Output

Bundle 1 (153 products): 1001, 2045, 3078, 4086, 5090...

Real-World Use

- Upselling at checkout
- Co-marketing recommendations
- Optimizing product shelving

5. Deliverable 2: Sensitivity Matrix via Influence Diffusion

Objective

Estimate how pricing changes in one product affect demand for neighboring products in the co-purchase graph.

Network Science Concept: *Edge Weight Diffusion + Influence Propagation*

By computing both direct and indirect influence via matrix operations, a **sensitivity matrix** was generated using:

$S=\beta A+\beta\alpha A^2$

- A: adjacency matrix
- β: scale of direct influence
- α: weight of indirect (2-hop) connections

Output Example

(0, 87048) 2.0 ← Strong link between products

(0, 249812) 1.5 ← Moderate influence

Real-World Use

- Dynamic pricing engine input
- Cross-elasticity estimation
- Discount propagation planning

6. Deliverable 3: Fault-Tolerant Recommender Design

Objective

Ensure recommendation continuity even when top-selling products are unavailable.

Network Science Concept: *Graph Robustness + Alternative Path Enumeration*

This module:

- Identified 1000 most central nodes using degree centrality
- For each central node, computed 2-hop backups
- Ranked backups by occurrence frequency and centrality

Example Output

Backups for product 524296: [112456, 174389, 245873, 334791, 100123]

Real-World Use

- Robust recommendations under stockouts
- Failure-resilient recommendation systems
- Reducing negative user experience

7. Deliverable 4: Centrality-Based Influence Propagation

Objective

Simulate how influential a product is in spreading interest through the network using cascade modeling.

Network Science Concepts:

- *Degree Centrality* – measures overall connectivity
- *Betweenness Centrality* – identifies bottleneck/influential nodes
- *Independent Cascade Model* – simulates influence spreading

Process Summary

- Graph built from reduceddataset14k.txt
- Top nodes selected by degree and betweenness
- Ran cascade diffusion model for each top betweenness node with:
    - Probability = 0.3
    - Max steps = 5

Example Output

Product 220345 drives adoption of 218 products:

{102938, 193847, 204958, ...}

Real-World Use

- Viral marketing simulation
- Strategic product placement
- Estimating word-of-mouth reach

Deliverable 5: Eigenvector Centrality-Based Influence Ranking

Objective

Identify the most influential products in the network based not only on direct connections (as with degree centrality), but also on the importance of their neighbors, using **Eigenvector Centrality**.

Methodology

This module processes the `reduceddataset.txt` graph to compute Eigenvector Centrality, which evaluates the recursive influence of a node within the network. In contrast to simpler metrics like degree or betweenness, Eigenvector Centrality assigns relative scores based on the centrality of a node's neighbors.

The steps are as follows:

1. **Graph Construction:**
   The dataset was loaded into a **directed graph** using NetworkX. Each undirected edge was treated as a pair of reciprocal directed edges to preserve mutual co-purchasing behavior.

2. **Centrality Computation:**
   NetworkX's `eigenvector_centrality` function was used, with:
    - Maximum iterations: 1000

3. **Ranking and Visualization:**
   The top 20 nodes by centrality score were extracted and visualized via a horizontal bar chart.

8. Summary of Deliverables

Module	Methodology	Real-World Application
Product Bundle Generator	Louvain Community Detection	Merchandising, upselling, co-marketing
Price Sensitivity Matrix	Influence Diffusion (Edge Weight Model)	Dynamic pricing, cross-product elasticity estimation
Fault-Tolerant Recommender	Graph Robustness + 2-Hop Failover Logic	Stockout mitigation, resilience planning
Influence Propagation Simulator	Centrality Metrics + Independent Cascade	Viral marketing, product seeding
Influence Ranking via Eigenvector	Eigenvector Centrality (Recursive Influence)	Strategic advertising, identifying authoritative nodes

9. Conclusion

This project demonstrates the potential of network science to solve real challenges in digital commerce. By modeling products as nodes and purchases as links, we implemented:

- Intelligent bundling
- Cross-price impact estimation
- Fail-safe recommendation engines
- Influence simulations for marketing planning

These tools serve as a foundation for smarter, more robust e-commerce platforms capable of adapting to user behavior and business constraints.