Supply Chain Data Transformation and Graphical Analysis using Neo4j

Assoc. Prof. Dr. Mervat Abuelkheir   Msc. Eng. Nada Labib   Eng. Ahmed Eloraby   Eng. Rawan Darwish

Eng. Mariam Wael   Eng. Mahmoud Nabil   Eng. Omar Galal   Eng. Hossam Magdy

***Abstract*—Enterprises are always faced with hard business decisions to make after a long process of supply chain analysis and a long pipeline of tools, staring from inputting data till extracting insights from the created visual models. This paper presents a comprehensive methodology for supply chain data analysis and graphical transformation into nodes and edges. Focused on enhancing decision-making and operational efficiency, the approach encompasses meticulous data preprocessing, innovative graph modeling, and advanced analytics. By leveraging tools such as the Neo4j graph database [3] [4] and Python libraries, the methodology delves into complex supply chain dynamics, including path finding, hidden link analysis, and product discontinuation impacts. In the face of evolving supply chain complexities, this framework provides valuable insights for businesses seeking to optimize their operations and navigate challenges effectively using only one tool. Scaling this project into an advanced framework for supply chain analysis and management by implementing more analysis tasks, highlighting future scalability within the realm of supply chain management.**

***Index Terms*—Supply Chain, neo4j, Data Analysis, Decision making, graph modeling, Knowledge Graph**

I. Introduction

Supply chain analysis involves a comprehensive assessment of each phase within a supply chain, commencing from the moment a business procures raw materials or resources from its suppliers to the ultimate delivery of finished products to customers [1]. The supply chain system involves various stages, including procurement, manufacturing, transportation, and distribution. The efficiency and effectiveness of a supply chain are crucial factors influencing a business's success, as they directly impact the timely delivery of products and overall customer satisfaction. In the face of evolving market dynamics and increasing global complexities, businesses are confronted with challenges in managing and optimizing their supply chains. That is why one of the critical challenges faced by most enterprises is initiating and overseeing the cycle of supply chain analysis as a whole and take momentous decisions regarding the supply chain. These decisions incorporate strategic choices in procurement, efficient management of inventory, optimization of transportation routes, and ensuring timely delivery while also even product cycle management.

According to the definition of a supply chain, we can envision a supply chain as a complex network encompassing all the activities involved in the production and delivery of a product [2]. Based on this, transforming this network into a graphical representation and translating the activities into visual elements can provide valuable insights and facilitate a deeper understanding of the entire process. By leveraging tools like Neo4j [4] for graphical analysis, we aim to create a comprehensive and dynamic visualization of the supply chain. This graphical representation goes beyond traditional methods, offering a more intuitive and interactive way to explore the relationships, dependencies, and flow of resources within the supply chain.

This approach enables stakeholders to identify bottlenecks, optimize workflows, and make informed decisions based on a holistic view of the entire supply chain. In essence, the transformation of the supply chain into a graphical representation allows for a more efficient and strategic approach to supply chain management [5].

The methodology employed in this study entails a systematic progression through various stages of supply chain analysis. Before the implementation phase, the initial steps were how to classify the supply chain network into nodes and edges was concluded after brainstorming on which members of the network behave as a node and which behave as an edge. Then a tool review was initiated to search and review for an eligible tool for transforming the supply chain network into a graph model. NetworkX [12] was one of the first candidates before Neo4j [4], however in comparison with Neo4j, all other tools fell short regarding the efficient usage of cypher query and interactive knowledge graph.

Starting with an internal data preprocessing, the focus lies on ensuring data integrity and readiness for subsequent transformations from data to dataframes. Subsequently, the transformation phase involves converting the data into a graph model from dataframes, effectively portraying the complex network structure inherent in the supply chain. A cost function is introduced, lending precision to the calculation of edge weights by considering essential pivotal attributes of the nodes, transportation costs, and distances between warehouses. The utilization of Neo4j as a graph database enhances the project's analytical capabilities, facilitating efficient representation and querying of the supply chain network [3]. Additionally, the methodology introduces the identification of critical nodes, hidden link analysis through clustering, and an examination of the ramifications associated with the discontinuation of a product within the supply chain.

II. Literature Review

This review was conducted to examine the dynamics of supply chain analysis in other papers, focusing on two distinct parts which are Supply Chain Analysis Tasks and Tools Utilized for Analysis. Trying to explore the landscape of supply chain evaluation, delineating the tasks involved and the diverse set of tools employed to navigate and analyze the complex network of operations and relationships of a supply chain.

A. Supply Chain Analysis

Multiple papers introduced types of supply chain analysis, of them is Supply chain vulnerability. Supply chain vulnerability involves identifying and mitigating potential disruptions, with resilience crucial for restoring service levels post-event. Traditional approaches may overlook downstream disruption risks, highlighting the need for a comprehensive understanding of supply chain structure. Various research methodologies, such as optimization and simulation, have been employed, but each has limitations. To address these gaps, Blackhurst et all [13] proposed an innovative approach combining Petri nets with the Triangularization Clustering Algorithm (TCA) to assess disruption vulnerability based on supply chain structure, mapping disruption propagation paths, providing a diagnostic tool for supply chain managers to evaluate and understand their networks' vulnerabilities.

Using knowledge graphs and disruptive event modeling, Ramzy et al. (2022) [14] presented MARE, a Semantic Supply Chain Disruption Management and Resilience Evaluation Framework. In order to assess and handle disruptions, MARE uses SPARQL-based recovery techniques and adds vital features like Strategic Stock, which is a pre-planned inventory used during disruptions, verified by inventory data and incurring warehousing costs. Alternative Shipment Strategy, which switches transport modes, queried through logistics data sources, revealing associated costs. Delayed Recovery, which checks partner capacity post-disruption to assess small delays for reduced financial losses, utilizing production data. Finally, Alternative Supplier Plan that seeks flexible secondary suppliers due to external disruptions, ensuring supply continuity albeit potentially at higher costs, sourced from procurement and supply management data. This framework tries to improve resilience and readiness in dynamic business environments by providing a forward-looking method for comprehending, planning, and recovering from supply chain disruptions.

B. Tools

While not an abundant amount of literatures incorporated Neo4j in their work, The paper [15] by W. Sun, Y. Li, L. Shi like in our proposed methodology adopts Neo4j to represent enterprises and logistics relationships as nodes and edges within the network. Using network theory, a graph model specific to an industry's supply chain network is formulated. The study introduces a supply chain network evaluation system encompassing both static and dynamic measurement indices, derived from statistical properties of complex networks. However, evaluations of the network's static and dynamic resilience are conducted through numeric experimental simulations for practical and theoretical insights for enterprises as proposed.

Centralizing the solution as one pipeline in one tool is the desirable objective in this paper, and this allows us to further introduce other supply chain related modules to create a full supply chain framework.

III. Methodology

A. Data Preprocessing

Preparing the data for transformation first involves data cleansing and preprocessing. The initial step is to examine the data for any missing values and substitute them with an arbitrary value like "Unknown" to mitigate their effects. Adding to that, when creating the dataframes, the naming convention of the data columns are adjusted using an English dictionary. Next, identifying missing relations between tables. For instance, we assume that all suppliers connected in the "Manufacturing" table ought to have at least one corresponding record in the "Shipments" table. To remedy this, we sought out all suppliers correlated in the "Manufacturing" Table but lacking any records linking them in the "Shipments" Table, after which we generate new records establishing those relations in the "Shipments" table. Moreover, We assume that each shipment must have at least one order. If a shipment lacks a record in the 'Orders' table, we add a new record with random values. We split shipments into four categories: internal SS shipments (between suppliers), internal SR shipments (suppliers to retailers), external supply chain shipments (suppliers to customers) and external RC shipments (retailers to customers). We also classify orders according to the shipments they belong to (same categories). To gain a better understanding of the data, we add new columns to multiple tables. We add a "products" column to the "Orders" table to specify which product is being ordered. We also add a "type of transportation" column to the "Shipments" table to indicate whether the shipment is being transported by air, land, or sea.

B. Transforming The Data From Relational model To Graph model

When the preprocessing stage is finished, the first step in transforming data into a graph database is to identify the nodes and edges. A graph database stores nodes and relationships instead of tables, or documents. Data is stored just like you might sketch ideas on a whiteboard. The data is stored without restricting it to a pre-defined model, allowing a very flexible way of thinking about and using it [3]. Nodes represent entities in the data, such as people, or things. Edges represent relationships between entities, such as Shipments. This is accomplished through a natural language processing model which analyzes the table names to determine whether they should be classified as nodes or edges. If this approach is not accurate, foreign keys within each table are examined, and the table is categorized into one of three groups based on the number of foreign keys it has: Nodes (0 foreign keys), Edges

(2 foreign keys), or Properties (otherwise). The properties' category is subsequently utilized as an attribute for the nodes or edges. Once it has been identified the nodes, edges, and properties, two tables are created, one for nodes and another for edges. in the nodes table, each record contains the name of its original table and all attributes it has. In the edges table, each record also contains the name of its original table as well as new attributes.

To perform the proposed Supply Chain Data Analysis and Transformation method, the first step is to perform the data transformation process which transforming the data into a graphical representation to visualize the all the Supply Chain paths. For starters, the original dataset is converted into two tables: a Nodes Table and an Edges Table. The Nodes Table will contain the Node ID, label, and other attributes required to define the node, while the Edges Table will include the From Node ID, To Node ID, and the order/service attributes needed to define the edge as properties. In the supply chain analysis project, fifteen different tables were utilized to map out the entire network. These tables represent different entities, transactions, and services involved in the supply chain, including *customers, external orders, external services, external shipments, external transactions, facilities, internal orders, internal services, internal shipments, internal transactions, manufacturing, products, retailers, suppliers, and warehouses.*

- **Customer Table:** Contains data related to customers, including their details and transaction history.
- **External Orders Table:** Records the orders placed by external entities, such as retailers or other suppliers, which then made in two separate tables.
- **External Services Table:** Details the services provided by external entities.
- **External Shipments Table:** Tracks shipments sent to external entities.
- **External Transactions:** Logs all transactions with external entities.
- **Facilities Table:** Provides details about the facilities used in the supply chain, such as warehouses and manufacturing plants.
- **Internal Orders Table:** Records the orders placed internally within the organization.
- **Internal Services Table:** Details the services provided internally within the organization.
- **Internal Shipments Table:** Tracks shipments sent internally within the organization.
- **Internal Transactions Table:** Logs all internal transactions.
- **Manufacturing Table:** Contains data about the manufacturing processes, including which suppliers are used for different products.
- **Products Table:** Provides details about the different products, including which warehouses they are stored in.
- **Retailer Table:** Contains data related to retailers, including their details and transaction history.
- **Supplier Table:** Provides details about the suppliers, including what products they supply.
- **Warehouses Table:** Contains data about the warehouses, including what types of products they store.

These tables are interrelated and interact with each other in various ways to form the complete picture of the supply chain network. For example, the Manufacturing table lists the different suppliers for each product, the Products table indicates which warehouses store each product, and the Warehouses table details the types of products stored in each warehouse.

By combining and analyzing the data from these tables, we can generate a comprehensive and detailed representation of the entire supply chain network, which can then be used for various analyses and optimizations.

After transformation, attributes are classified as nodes and edges having approximately 5900 nodes and 8120 edges.

C. Calculation of Edge Weights using the Cost Function

In this context, the cost function is a tool designed to calculate the weights of the edges in the graph. This graph in focus represents a network of suppliers, products, warehouses and retailers. The weight of an edge essentially signifies the "cost" associated with that edge. A lower weight indicates a more desirable or "less costly" edge.

The process of calculating the cost function involves several key steps:

1) Initialization: The function first verifies that the data frames for edges and nodes are not empty. If they are, the function will not continue.

2) Baseline Determination: The function computes the averages for several key parameters in the focus group, including annual sales, market share, price, profit margin, and rental price. These averages serve as a baseline for comparison in the next step.

3) Preliminary Weight Calculation: For every edge in the graph, weight is initialized as a value of 100, the function evaluates the attributes of the connected nodes. It specifically targets the cost-related attributes (annual sales, market share, price, profit margin, and rental price). The function then compares each attribute's value to its corresponding average value determined in the previous step. Depending on whether the attribute's value is greater or less than the average, the function subtracts a specific amount from the edge's preliminary weight.

4) Handling Missing Values: After processing all the edges, the function fills any missing values in the edge and node data frames with zeros.

5) Transportation Cost Determination: The function calculates a transportation cost for each edge; a web scraper will be used to gather the distance and duration of the transportation of a product between each two suppliers in the graph from a real-time website. The distances and durations will be normalized between the range 0 and 100 using the min-max normalization technique. The final transportation cost will be the average of the edge's distance and duration. This transportation cost is then subtracted from the edge's preliminary weight.

6) Weight Range Normalization: The function adjusts the weights of the edges to ensure they fall within a specific range (e.g., 1 to 100).

7) Final Weight Adjustment for Warehouse-Supplier Edges: The function further refines the weights of the edges based on their distance. If the distance falls within certain thresholds, a particular amount is subtracted from the weight.

*8) **Final Cost Equation**:* Upon completion of these steps, the final weight (**Weight**) for an edge, which symbolizes its cost, is calculated as follows:

$$\begin{aligned} \text{Weight} = 100 &- \text{Attribute Deviation Adjustment} \\ &- \text{Transportation Cost} - \text{Warehouse Distance Cost} \end{aligned} \tag{1}$$

In this equation:

- **Attribute Deviation Adjustment** is the total amount subtracted based on how much the node attribute values deviate from their average values. The specific amount subtracted for each attribute is predefined in the cost function.
- **Transportation Cost** represents the average of the transportation distance and duration associated with the edge.
- **Warehouse Distance Cost** is the amount subtracted based on the distance between the warehouse and the supplier. The specific amount subtracted for each distance range is predefined in the cost function.

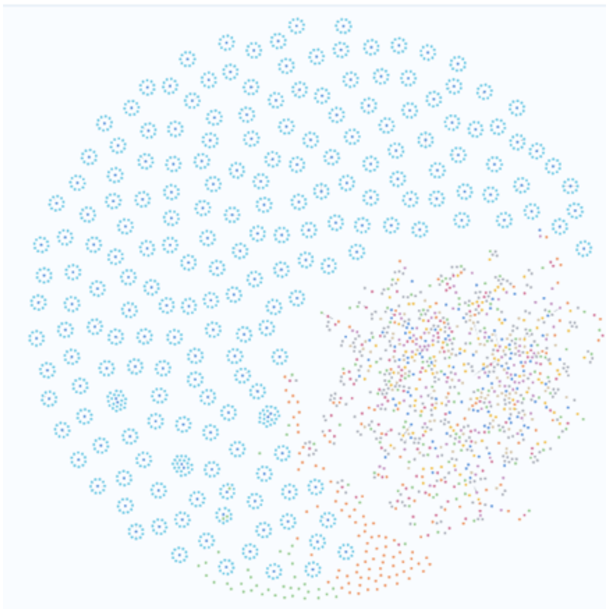
It's important to note that the Attribute Deviation Adjustment, Transportation Cost, and Warehouse Distance Cost could potentially cause the Weight to fall below 0. In such a case, the weight should be reset to a minimum value (e.g., 0 or 1) to avoid negative weights. Also, rental price baseline was calculated initially in the Preliminary Weight Calculation step, so it will not be added again in the warehouse cost as the focus was the distance cost between the supplier and the warehouse.

This cost function thus takes into account multiple factors to compute the edge weights, providing a comprehensive measure of the "cost" associated with each edge in the network.

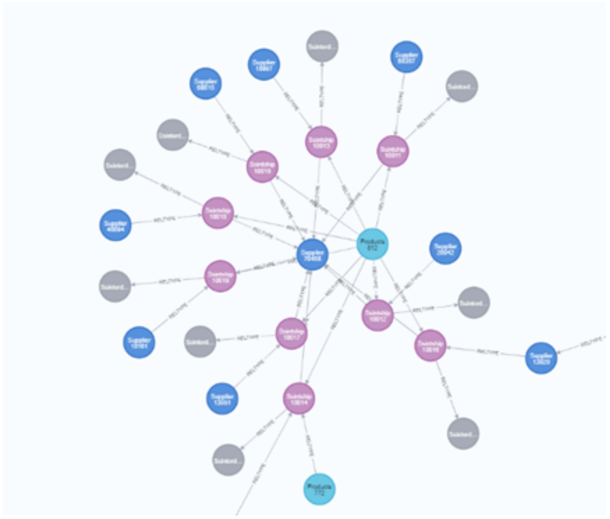
Finally, the two tables are passed to a graph query program, Neo4j. Neo4j is a graph database that allows for the storage and querying of large and complex networks of data, and is able to interpret and visualize this data as a supply chain network graph [4].

Neo4j takes the two tables, one representing the nodes (or entities) and the other detailing the edges (or relationships) and constructs a graph that models the supply chain network. Each node could represent a supplier, warehouse, or product, and each edge could represent the flow of goods, the transportation route, or any other interaction between these entities. The edge weights, calculated using the weight cost function, are incorporated into the graph as properties of the edges.

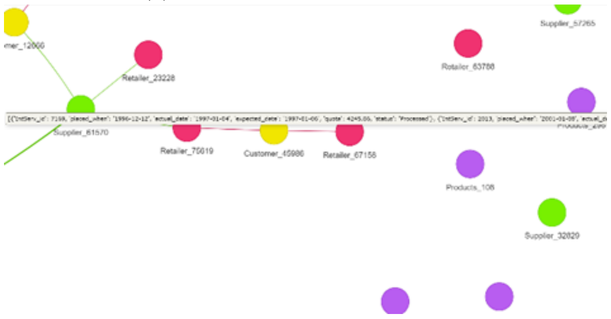
One of the key features of Neo4j is its powerful graph querying language, Cypher, which allows for efficient and sophisticated querying of the graph. This could involve finding the most cost-effective route between two entities, identifying



(a) Neo4j's output illustrating the supply chain as a whole network.



(b) A closer look at the network.



```
[{'intServ_id': 7108, 'placed_when': '1996-12-12', 'actual_date': '1997-01-04', 'expected_date': '1997-01-06', 'quota': 4245.86, 'status': 'Processed'}, {'intServ_id': 2013, 'placed_when': '2001-01-08', 'actual_d
```

(c) Interaction between a supplier and a retailer.

Fig. 1: Figure 1a shows Neo4j output illustrating the supply chain as a whole network, while Figure 1b and Figure 1c take a closer look at the network. Figure 1c highlights the interaction between a supplier and a retailer. The connecting edge represents internal services, encapsulating key attributes of the operations. These attributes could include service frequency (quota), expected arrival date, and status, providing a snapshot of the supply chain dynamics between the two nodes.

key suppliers or products based on their connections, or uncovering potential bottlenecks in the supply chain.

Furthermore, Neo4j's graph visualization capabilities make it easy to understand and analyze the structure of the supply chain network. Nodes and edges can be color-coded based on their properties, making it straightforward to visually identify high-cost paths, central entities, and other features of interest.

In this way, Neo4j takes the complex, multidimensional data of a supply chain and transforms it into a clear, intuitive, and interactive graph. This graph not only represents the structure of the supply chain but also encodes the costs associated with different paths and interactions, making it a powerful tool for supply chain analysis and optimization.

Eventually, this methodology will provide a comprehensive analysis of the supply chain network, allowing for informed decisions to be made based on the calculated cost between nodes.

D. Finding Paths for Graphical Analysis

To be able to perform graphical analysis on our dataset, we first had to find a way to retrieve all or some paths, ideally with the least cost, between given nodes that were to be analyzed later. This was made possible by using Neo4j's Dijkstra's algorithms. Neo4j offered three variations of the algorithm. The first being Dijkstra Single-Source Shortest Path, which allows us to retrieve the shortest path from a specified node "n" to every other node it is connected to. It also supported both directed and undirected graphs, as well as weighted graphs. The second variation was Dijkstra Source-Target Shortest Path, which was able to retrieve the shortest path between two specified nodes "n" and "m" if it exists. Or in the case of weighted graphs, it would return the path with the least cost. Lastly, Yen's algorithm Shortest Path was used to find "k" the shortest paths between two specified nodes "n" and "m". The easier task was to find paths that go with the natural sequence of the hierarchy of a supply chain, ie: from a "supplier" node to a "customer" or a "retailer" node. However, the challenge was to find the needed paths that go in the opposite direction efficiently, especially that the graph was directed.

E. Critical Nodes

Identifying risks is a crucial part of the survival of any business. If identified early, it could help create back-up plans or alternative solutions. This was exactly the motive behind this task. We wanted to identify critical nodes that qualify as risks in the long run to a certain business or node "n". Most nodes shared some common features that could hint to its criticality such as: the in or out degree of a node, its geographical location, the kind of product being sold, manufactured, shipped or stored, the price and weight of a product. The degree of a node hints at how high or low the demand for a certain supplier/retailer/product or generally any node is. Similarly, nodes could be considered critical if they were geographically isolated in a region of their own. Cutting off relations with such a node could mean losing out on a whole region, whether that being losing out on customers in the region or affordable labor force and resources. At any stage of the supply chain, a bad decision could be costly. Most importantly, product attributes could have severe effects on a whole business. The product itself could be monopolized by one or very few suppliers and therefore deemed critical. In such cases a business must become aware of this early on to either strengthen ties with the supplier(s), find alternative solutions or even close down before major losses. The price and weight of the product also plays a tremendous role in business. Heavier products cost more to ship, especially from further locations. As well as finding the cheapest option even by a small margin could prove to be beneficial in the long term. A special case was made for nodes that represent Warehouses, as it was observed that their criticality was determined by certain features that are specific to them. It was found that the percentage of a specific product as well as the quantity of it that is stored in each warehouse could hint at some being critical. It is also worth mentioning that the geographical location of each warehouse and cost they charge played a role, but these were covered with the rest of the cases previously. If a warehouse was to hold all of a product (in some cases all of it in just one region) then it should have high criticality. But even if the product is not being monopolized by just one warehouse but either a majority of the product is being stored there or the quantity of it was to the extent where it is not possible to replace it by just one of the current available warehouses (keeping in consideration that they are also serving other clients) it will be considered critical. In order to be able to implement these strategies, the dataset had to be modified. This was possible because the original dataset was auto-generated, so there was no problem with data accuracy. A column named "warehouses" already existed in the table "products" which was composed of a list of id of warehouses that store a certain product "p". Therefore, a new column was created in the same table named "capacity" which stored a list of the same size as in the column "warehouses" that holds the quantity of the respective product "p" in each of the warehouses specified in the "warehouses" list. The values were randomly generated based on the capacity of each warehouse.

F. Hidden Link Analysis

This study employed a hidden links' analysis through clustering as a means of identifying potential hidden relationships between nodes in a supply chain that may not be readily apparent in the available data. The underlying hypothesis behind this approach is that nodes sharing similar attributes, such as geographic location, product type, or customer base, are likely to exhibit comparable supply chain behavior. Clustering, as a technique for grouping similar nodes, facilitates the identification of such groups in the supply chain. Through the use of clustering, the study identified groups of nodes with similar characteristics that could potentially lead to diverse paths within the supply chain, by having multiple destination nodes with comparable attributes. By identifying these hidden relationships, it is possible to optimize the supply chain through informed decision-making by switching to these

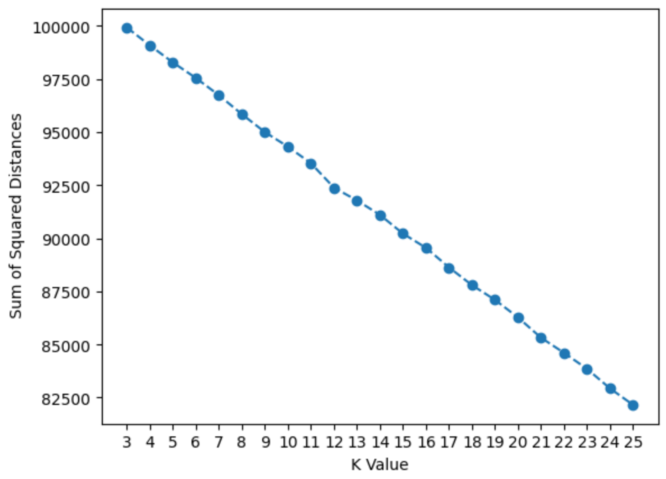


Fig. 2: Elbow method for finding optimal K (number of clusters) to be K=12.

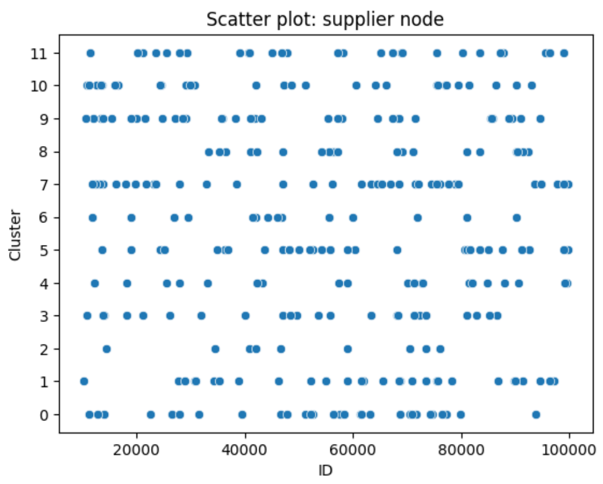


Fig. 3: Each supplier against his cluster

hidden paths and utilizing them, which can help improve supply chain efficiency and reduce associated costs. Python version 3.8.10 [6] and various libraries, including NumPy [7], pandas [8], matplotlib [9], seaborn [10], and sklearn [11] were used to conduct this analysis.

- Dataset Preparation: The ReadingDataSet module reads the dataset and creates a dictionary containing all data frames, then identifies unique labels for different nodes and creates separate datasets for each node label.
- Exploratory Data Analysis (EDA): An initial exploratory data analysis was performed on the "customer_node" since it is the last node in the chain. This node will be utilized in the elbow method to choose a suitable number of clusters (K). This included generating a histogram showing the distribution of 'age' with respect to 'gender'.
- Preprocessing: The data is further preprocessed to remove unwanted columns, such as those with 'id' in their names and "Label." Every linear data in the tables were used in the analysis. The data types of each element in the nodes were also determined, and columns with non-linear data types were removed before clustering. StandardScaler and OneHotEncoder were used for feature scaling and encoding categorical variables, respectively.
- KMeans Clustering: The KMeans clustering algorithm was applied to a range of cluster sizes (from 3 to 26, this range was applied to large data points) to find an optimal cluster size. The elbow plot and the rate of change of SSD helped identify $k = 12$ as the optimal number of clusters as shown in figure (2).
- Node-Wise Clustering: The code implements clustering on all nodes by first separating node-wise data according to type (scaling for numerical data and encoding for categorical data), and then applying the KMeans model. This resulted in assigning each data point to a specific cluster for all nodes.
- Visualization: Scatter plots were used to visualize the clustering results, plotting the supplier "ID" against the "market_share (%)" while showing the suppliers in the same cluster with the same hue color.

The results can be further improved by involving a valid data that reflects real life scenarios of a supply chain. Moreover, interpreting the results from the cluster analysis, identifying characteristics of each cluster, and understanding the relationships or patterns within the data that lead to these groupings. Additionally, identifying outlier nodes (disregarding irrelevant nodes) or anomalies in the clustering results could provide more insight into unique supply chain dynamics or potential areas for improvement.

G. *Results*

The results are plotted in the following figures, having the X-axis as ID (id of each node) and Y-axis as Cluster (cluster of each node belongs to) as shown in figures (3) (4) (5) (7) (8). Nodes on the same horizontal line are on the same cluster since they share similar properties, therefore a source node can to switch to a different new destination node that belongs to the same cluster as the old destination node, creating a new path.

H. *Product Discontinuation Analysis*

In the Product Discontinue Analysis, our model was utilized to estimate the effects of a supplier discontinuing a product on both the upper and lower layers of their supply chain. Our method involved determining the ratio between the manufacturing cost of the product in question and the total manufacturing cost of all other products. Additionally, we made the assumption that the cost of orders placed by the supplier for manufacturing each product is divided among the products based on their respective manufacturing costs. By employing this approach, we were able to assess the consequences of a supplier discontinuing a product they manufacture. Specifically, we calculated the ratio mentioned

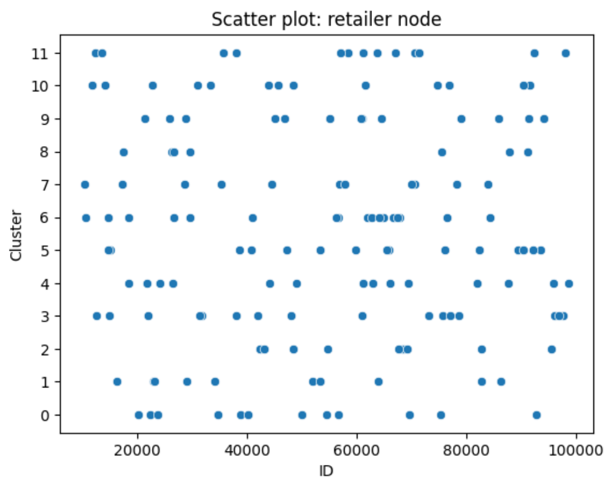


Fig. 4: Each retailer against his cluster

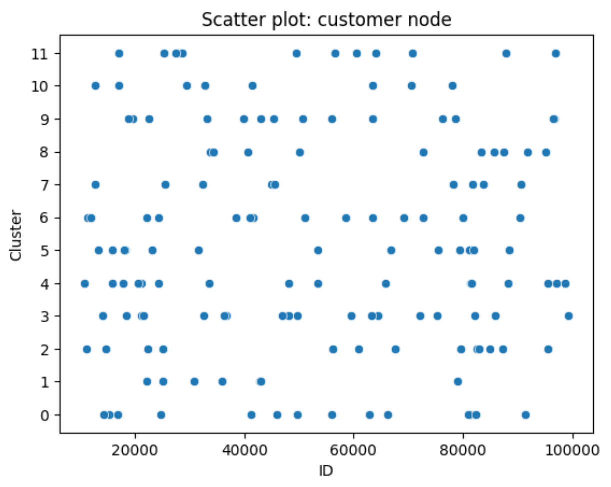


Fig. 5: Each customer against his cluster

earlier and applied it to analyze the orders placed by this supplier with other suppliers. Our analysis indicated that the supplier discontinuing the product would experience a gain equivalent to the total costs of these orders multiplied by the ratio, while the suppliers fulfilling these orders would suffer a loss equal to the cost of the order multiplied by the same ratio.

IV. Conclusion

In summary, the very complicated workings of modern supply chains demand a thorough understanding and management approach. From sourcing materials to delivering finished products, a seamless supply chain is crucial for any business's success. However, businesses continually face challenges in adapting to market changes and global complexities, highlighting the ongoing need for effective supply chain analysis.

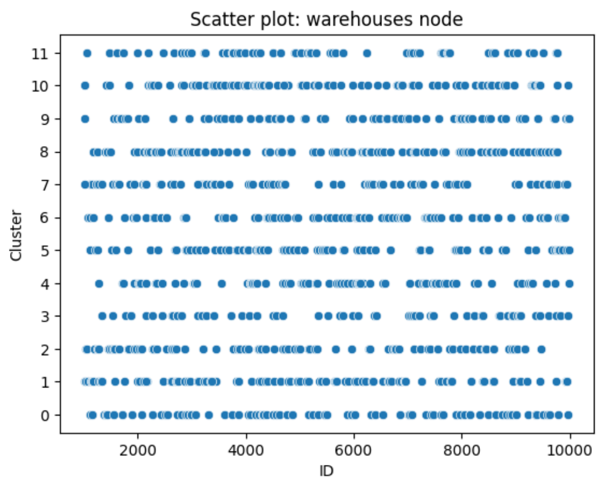


Fig. 6: Each warehouse against its cluster

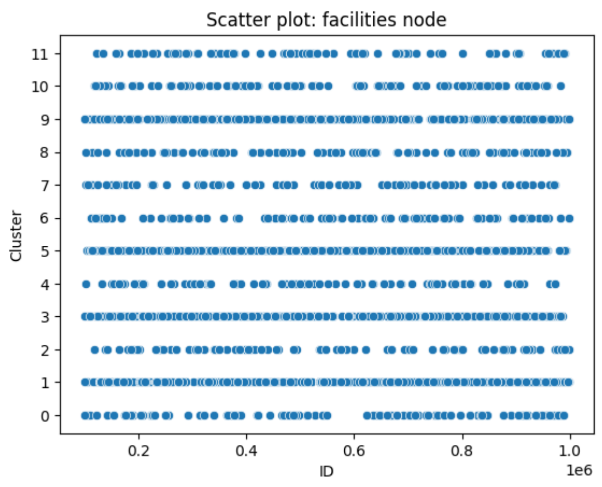


Fig. 7: Each facility against its cluster

The study emphasizes the importance of visualizing the supply chain as a complex network. Using tools like Neo4j moves beyond traditional methods to create intuitive, interactive representations that reveal the intricate connections, dependencies, and resource flow within the supply chain. This visual approach empowers decision-makers to spot bottlenecks, streamline processes, and make well-informed choices.

The methodology involved several key steps. It began by categorizing the supply chain into nodes and edges, selecting Neo4j over other tools due to its superior ability to transform networks into graphical models. Following data preprocessing and transformation, a precise cost function was introduced, leveraging Neo4j's capabilities for effective representation and analysis.

Notably, the approach goes deeper into pinpointing critical nodes, unraveling hidden links through clustering, and exploring the ripple effects of discontinuing a product within

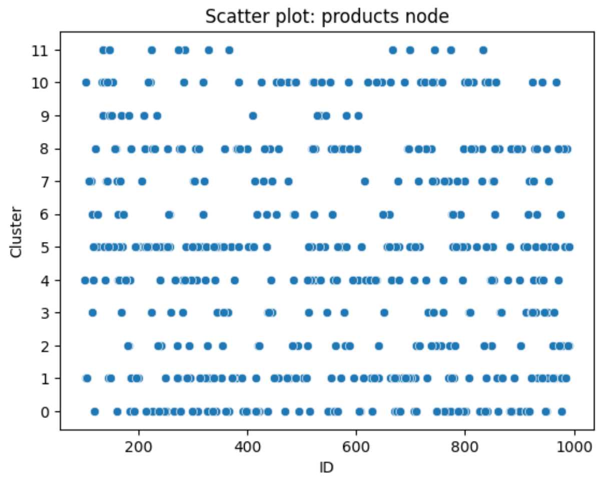


Fig. 8: Each product against its cluster

the supply chain. Embracing graphical representations and Neo4j's functionalities led to the adoption of a strategic and efficient approach to supply chain analysis. This paves the way for smarter decision-making and more effective management strategies in optimizing supply chain operations.

V. Future Work

The ambition of this project is to apply this analysis pipeline on any similar complex data, to have the generalized form of this pipeline to handle other case problems that are close in network shape complexity and analysis complexity, for instance social media analysis. At this point we can unlock the hidden powers of this tool, then implement for task specific evaluations and resilience.

There will be additional improvements to the cost function to make it more complex for handling more realistic scenarios.

Furthermore, introducing the ability to manually enter the data from a user-friendly interface. Expanding the project's scope involves exploring avenues for enhanced user interaction and decision-making so to make the system suited for enterprises. One of the make future goals is to integrate additional artificial intelligence algorithms in the system to predict more supply chain disruptions and recommend proactive accurate measures. Nevertheless, adding more analysis tasks referencing the reviewed papers to confirm the tasks practical and relevant for the business owners.A phase in hidden link analysis involves leveraging Neo4j link predictor algorithms, offering a clear advantage over clustering methods in terms of efficiency and enabling a smoother execution process.

References

[1] IBM. *What is supply chain analytics?*, https://www.ibm.com/topics/supply-chain-analytics

[2] Investopedia. (n.d.).*The Supply Chain: From Raw Materials to Order Fulfillment*,https://www.investopedia.com/terms/s/supplychain.asp

[3] Neo4j. *What is a Graph Database?*,https://neo4j.com/developer/graph-database/

[4] Neo4j. *Neo4j Graph Database*, https://neo4j.com/

[5] Wikipedia. *Supply Chain Management*, https://en.wikipedia.org/wiki/Supply_chain_management

[6] Python Software Foundation. (2023). *Python 3.8.10 Documentation*. https://www.python.org/downloads/release/python-3810/

[7] Harris, C.R., Millman, K.J., van der Walt, S.J., et al. (2020). *Array programming with NumPy*. https://numpy.org/doc/stable/

[8] McKinney, W. (2023). *pandas: Powerful data analysis tools for Python*. https://pandas.pydata.org/docs/

[9] Hunter, J.D. (2022). *Matplotlib: Visualization with Python*. https://matplotlib.org/stable/contents.html

[10] Waskom, M. et al. (2023). *Seaborn: Statistical Data Visualization*. https://seaborn.pydata.org/

[11] Pedregosa, F. et al. (2023). *scikit-learn: Machine Learning in Python*. https://scikit-learn.org/stable/

[12] NetworkX Documentation. (n.d.). https://networkx.org/documentation/stable/tutorial.html

[13] Jennifer Blackhurst, M. Johnny Rungtusanatham, Kevin Scheibe, Saurabh Ambulkar, Supply chain vulnerability assessment: A network based visualization and clustering analysis approach, Journal of Purchasing and Supply Management, Volume 24, Issue 1, 2018, Pages 21-30, ISSN 1478-4092, https://doi.org/10.1016/j.pursup.2017.10.004.

[14] *MARE: Semantic Supply Chain Disruption Management and Resilience Evaluation Framework*,
Nour Ramzy, Soren Auer, Hans Ehm, Javad Chamanara,
arXiv preprint **2205.06499** (2022),
https://arxiv.org/abs/2205.06499,
Primary Class: cs.DB.

[15] W. Sun, Y. Li, L. Shi, *The Performance Evaluation and Resilience Analysis of Supply Chain Based on Logistics Network*, in *2020 39th Chinese Control Conference (CCC)*, pp. 5772-5777, 2020. doi: 10.23919/CCC50068.2020.9189234.