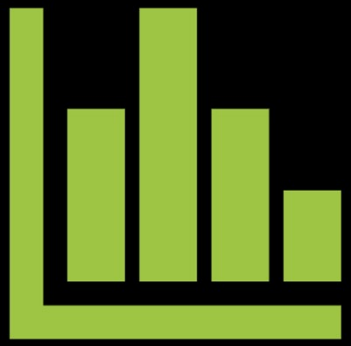
INSPECTING THE MAIN CHALLENGES FACED IN BIG DATA ANALYTICS TOOLS DURING VISUALIZATION:

A STUDY



Jaichiddharth Rangasamy Karthigeyan

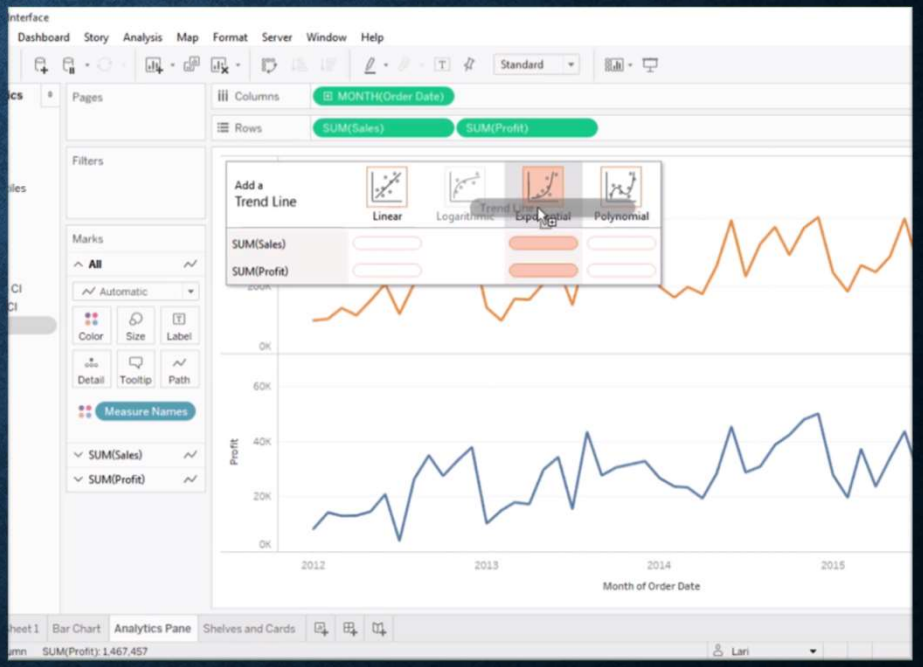
A20527281

OVERVIEW

- Data visualization tools like Power BI, Tableau, Plotly, and Gephi are essential for extracting insights from data and making informed decisions. However, these tools face challenges with parallel processing, operating system limitations, and input/output bottlenecks. Existing remedies include data optimization, network optimization, hardware optimization, and parallel processing, but new approaches are needed.
- The study is performed to examine the pros and cons associated with these tools. It helps in knowing the parallel processing issues and i/o bottleneck and the mechanisms used to overcome the same.
- Moreover, the role of Hermes is mentioned as possible hypothesis to make an assumption that how far it would further reduce the i/o bottleneck. However, at this time, there is no indication that Hermes is currently used in these tools.

TABLEAU

- Tableau desktop: It is a self-service analytics and visualization tool used by analysts and data scientists to build and publish interactive dashboards, reports, and visualizations. Tableau server is a web-based platform.
- In order to extract data and analyze it in memory, Tableau's **data engine** establishes a connection with the data source. These speeds up the analytical process.
- Spreadsheets, databases, cloud-based data warehouses, and web-based data are just a few of the many **data sources** that Tableau supports.



PARALLEL PROCESSING IN TABLEAU

- Tableau has added a tool called **“Hyper”,** an **in-memory** data engine that permits parallel processing, to address this issue. Hyper makes use of the multi-core CPUs and GPUs' processing capacity to help Tableau manage bigger datasets and perform better during visualization.

- Tableau uses data partitioning and parallel aggregation for larger datasets to reduce processing time, multithreading and graphics acceleration for rendering visualizations.

- Tableau also has a function called "**Data Extracts**" that enables data to be processed and aggregated **outside of Tableau**. The processing burden on Tableau may be lessened, and visualization performance can be enhanced.

OPERATING SYSTEM AND TABLEAU RELATIONSHIP

- Problems with memory allocation, file management, and interoperability with various operating systems are frequently encountered.
- "**Tableau Server**", is used to get around these difficulties. In order to manage bigger datasets and enhance visualization performance, it provides **scalability**.
- "**Tableau Prep**" enables data to be cleaned up and modified before being viewed in Tableau which **reduces** processing burden and **improves** visualization performance.

CAUSES OF TABLEAU'S INPUT/OUTPUT BOTTLENECK

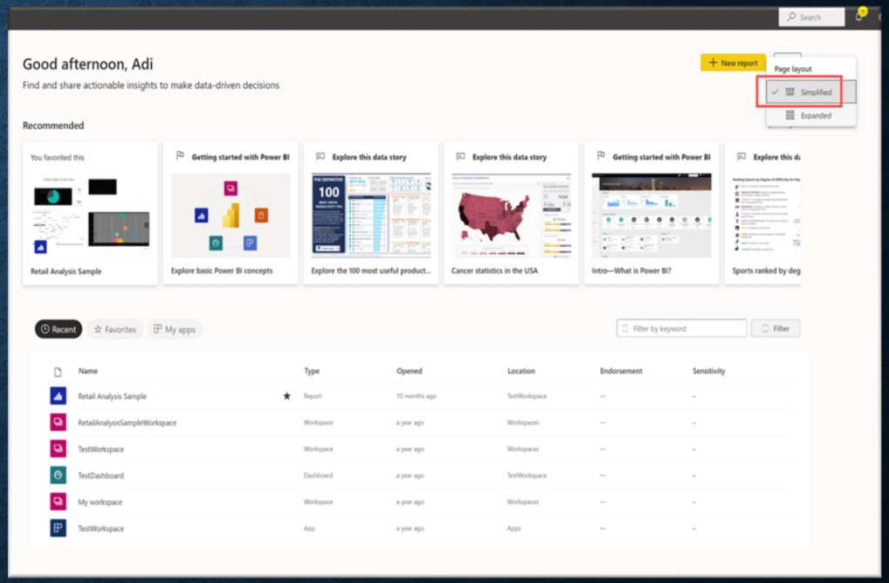
- Performance of Data Sources: Tableau wouldn't be able to get the data rapidly if the data source had a delayed response time, creating an input/output bottleneck.
- Data Volume: Tableau processes enormous amounts of data, and if the volume of data **exceeds** the I/O subsystem's capabilities, it may result in an input/output bottleneck. This is especially accurate for big data extraction.
- Data must be communicated across the network since Tableau Server is often located on a different workstation than the data source. Data retrieval and analysis might be delayed as a result of network latency, creating an input/output bottleneck.
- Disk performance: When processing data, Tableau writes and reads temporary files. If the disk performance is slow, this might cause an input/output bottleneck.

OVERCOMING INPUT/OUTPUT BOTTLENECK IN TABLEAU

- Data source optimization involves making the most of efficient **indexing and searches**.
- Reduce the quantity of data that must be exchanged between the data source and Tableau by using **data extracts**.
- Network optimization: Make the Tableau Server and the data source's network connections as efficient as possible. To speed up data transfer, use a high-speed network connection with minimal latency.
- Hardware Optimization: Utilize a high-performance CPU and RAM, high-performance drives and optimize disk performance parameters to improve hardware performance to process huge data quantities.

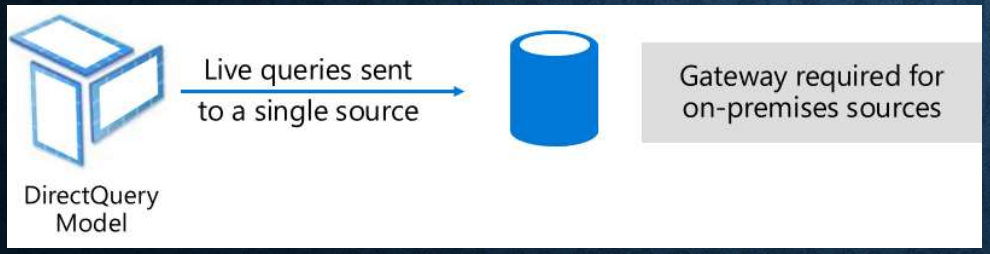
POWER BI

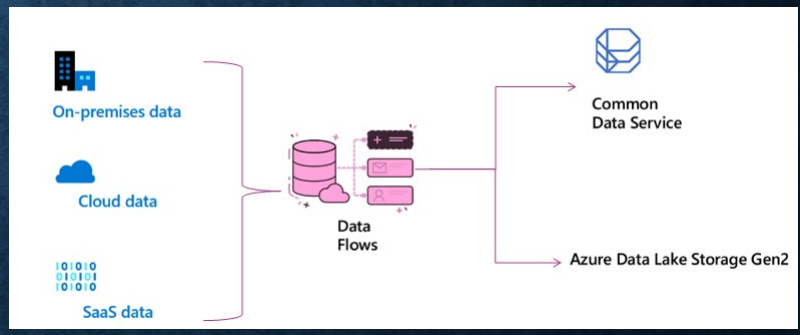
- Users of Power BI Desktop may generate interactive visualizations and reports as well as connect to a variety of data sources and build data models using a drag-and-drop interface.
- Power BI Mobile is a mobile app that can be used on iOS, Android, and Windows devices to view dashboards and reports made in Power BI Desktop and Power BI Service from any place.
- In order to extract data, execute data transformation, and generate data models, Power BI's data engine establishes a connection with the data source.
- Excel spreadsheets, cloud-based data warehouses, databases, and web-based data are just a few of the many data sources that Power BI can connect to.



PARALLEL PROCESSING IN POWER BI

- As Power BI calls for particular abilities and expertise, integrating parallel processing in visualization is a difficult undertaking.
- Instead of having to load data into Power BI's memory, "**Direct Query**", enables data to be queried straight from the source database. Direct Query makes use of the **processing capacity** of the underlying database to handle bigger datasets and perform better during visualization.

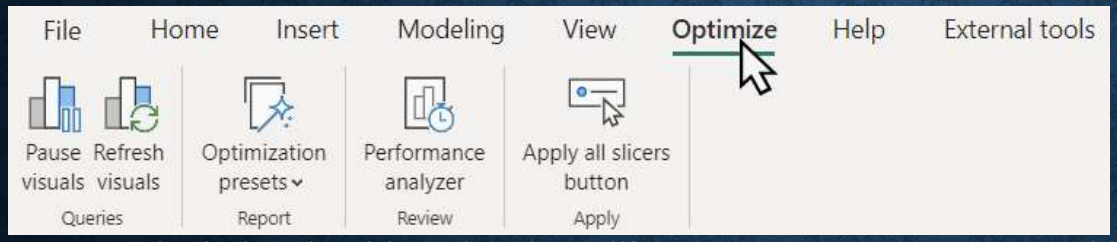




- A component of Power BI called **"Dataflows"** also enables data to be prepared and converted outside of Power BI so that processing burden may be reduced, and visualization performance can be enhanced.

OPERATING SYSTEM AND POWER BI RELATIONSHIP

- Power BI offers a tool called "**Power BI Desktop Optimization**," which enhances Power BI Desktop's functionality on Windows. This feature provides decrease in the number of files kept on the hard drive and an increase in the amount of RAM that Power BI has access to.



- Power BI includes cloud-based options, like as "**Power BI Premium**" which has a dedicated processing and rendering engine to offer a uniform platform for data analysis and visualization while assisting in overcoming operating system problems.
- Tableau uses data partitioning and parallel aggregation for larger datasets to reduce processing time, multithreading and graphics acceleration for rendering visualizations.

CAUSES OF POWER BI'S INPUT/OUTPUT BOTTLENECK

- Data Volume: Power BI analyzes enormous amounts of data, and if the volume of data exceeds the I/O subsystem's capabilities, it may result in an input/output bottleneck. This is especially accurate for big data extraction.
- Performance of Data Sources: Power BI can connect to a variety of data sources, including databases, flat files, and cloud-based data storage. Power BI wouldn't be able to swiftly obtain the data if the data source had a delayed response time, creating an input/output bottleneck.
- Network Latency: Data must be delivered across the network since Power BI is often housed on a different workstation than the data source. Data retrieval and analysis might be delayed as a result of network latency, creating an input/output bottleneck.
- Disk performance: When processing data, Power BI writes and reads temporary files. If the disk performance is sluggish, this might cause input/output bottlenecks.

OVERCOMING POWER BI'S INPUT/OUTPUT BOTTLENECK

- Data source optimization involves making the most of efficient indexing and searches.
-
- Data Model Optimization: Reduce the number of tables, columns, and connections in the data model and aggregate the data at a higher level to improve it. Performance can be enhanced, and the volume of data can be decreased.

- Network Optimization: Improve the network connection between the data source and the Power BI Service. To speed up data transfer, use a high-speed network connection with minimal latency.

- Hardware Optimization: Utilize high-performance drives and optimize disk performance parameters to improve hardware performance. Make sure Power BI has adequate resources to handle massive data volumes by using a high-performance CPU and RAM.

PLOTLY

- Plotly Graphing Library: The power to produce and present interactive plots and visualizations in several computer languages, such as Python, R, and JavaScript, is provided by this core package. To improve user engagement with the data, it also has tools like zooming, panning, hovering, and filtering.

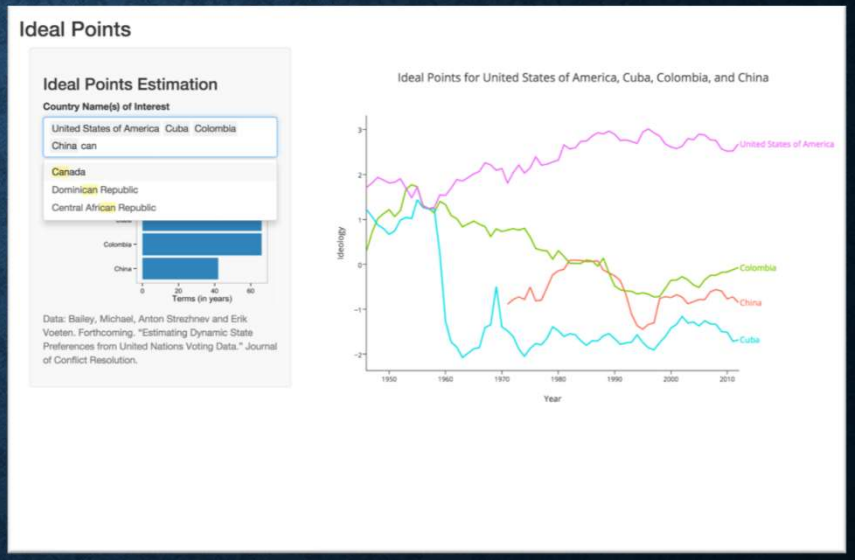
- Plotly Chart Studio: This web application offers a platform for sharing and creating charts, dashboards, and reports that were made using Plotly. In Chart Studio, users may use a drag-and-drop interface to build and tweak visualizations before sharing them with others through URL or embedding them in web sites.

- Plotly Dash: With the help of Plotly visualizations, users may build interactive web apps utilizing this Python-based web platform. Users of Dash may create unique dashboards and data-driven web apps with cutting-edge features including dynamic layout, real-time data updates, and interactive controls. It smoothly interacts with Plotly Graphing Library and makes use of Flask as the foundational web framework.

PARALLEL PROCESSING IN PLOTLY

- Plotly employs parallel processing techniques to speed up the development of charts. When plotting huge datasets, users could encounter performance issues since the library might not be utilizing all of the computing capabilities. Users can speed up the production of charts by using multiprocessing, a Python tool that enables parallel processing, to remedy this issue.

- Plotly's most recent version (v5) now supports data partitioning, multi-processing and multi-threading in addition to increased parallel processing features. This enables users to make advantage of additional computing power when making and modifying plots with huge datasets, leading to visualizations that are quicker and more effective.



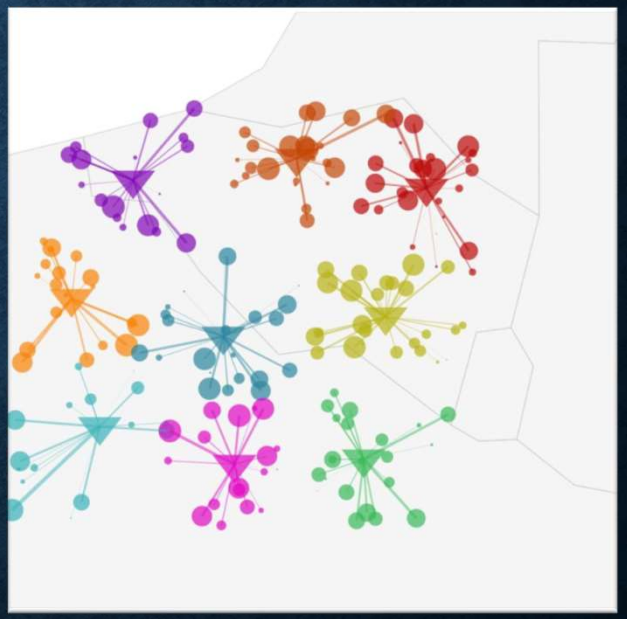
OPERATING SYSTEM AND PLOTLY RELATIONSHIP

- Plotly is a web-based platform that can be used on any operating system that supports a web browser, hence there are no operating system-specific problems. However, customers using specific web browsers, especially older ones, may run into compatibility difficulties. It is advised to use the most recent version of a compatible web browser to prevent compatibility problems.

- Plotly v5 now works better with a wider range of web browsers and operating systems. To guarantee that users may build and see plots on their favorite browser without any compatibility difficulties, the library has been tested and optimized for the most recent versions of well-known web browsers like Chrome, Firefox, and Safari.

CAUSES OF PLOTLY'S INPUT/OUTPUT BOTTLENECK

- Data Volume: Plotly's input/output bottleneck is frequently caused by the amount of data being handled. The reading and writing of large datasets might take a long time, which can slow down the visualization.
- Network delay: Network delay can be a significant obstacle when using Plotly to display data from a remote server. Long load times and sluggish visualization might result from slow network connections.



SOLUTIONS TO OVERCOME CHALLENGES IN PLOTLY

- Data optimization: Make the most of your data by employing effective indexing and searches. This can speed up the visualization process and lower the amount of data that must be processed.
- Network optimization: Improve the server's and the visualization tool's connections to the internet. To speed up data transfer, use a high-speed network connection with minimal latency.
- Utilize high-performance drives and optimize disk performance parameters to improve hardware performance. Make sure Plotly has adequate resources to handle big data volumes by using a high-performance CPU and RAM.

GEPHI

- Graph Model: Bipartite graphs, weighted and unweighted graphs, directed and undirected graphs, and other types are all supported by Gephi. The graph model also has attributes for edge color, weight, and labeling as well as node size, color, and labeling.

- Gephi Desktop: Users of Gephi Desktop may import data from several sources, such as databases and CSV files, and see the data as a network graph. Users may filter the graph based on node or edge qualities, organize the nodes and edges of the network using layout algorithms, zoom, pan, and pick nodes and edges to interact with the graph.

- Gephi Toolkit: This library, which is Java-based and offers a programmatic interface to the Gephi engine, enables programmers to build unique applications that make use of Gephi's visualization and exploration features. Modules for data import/export, graph manipulation, layout methods, and visualization are all included in the Gephi Toolkit.

PARALLEL PROCESSING IN GEPHI

- Gephi enables users to see and examine massive networks and graphs. The program employs "multi-threading," a parallel processing method, to expedite the construction and modification of graphs. However, when dealing with big graphs that demand more processing power than is available on their device, users may encounter performance issues.

- Users can leverage Gephi's built-in parallel processing features, such as the "Progressive Layout" feature, which enables the gradual arrangement of big graphs, to solve this problem.

- The most recent version of Gephi (v0.9.2) provides a number of enhancements to its parallel processing capabilities, such as better data partitioning, multi-threading support and memory management. As a result, users can work more effectively and efficiently with larger and more complicated graphs.

OPERATING SYSTEM AND GEPHI RELATIONSHIP

- Gephi was created in Java and may be used on any platform that supports Java. When utilizing obsolete operating systems or earlier versions of Java, users might run into performance problems. It is advised to use the most recent Java version and a suitable operating system to prevent performance difficulties.

- Improved interoperability with more recent Java releases and a variety of operating systems is included in Gephi version 0.9.2. Users may use the program on their favorite operating system without experiencing any performance difficulties because it has been tested and optimized for the most recent Java versions as well as several widely used operating systems including Windows, MacOS, and Linux.

CAUSES OF GEPHI'S INPUT/OUTPUT BOTTLENECK

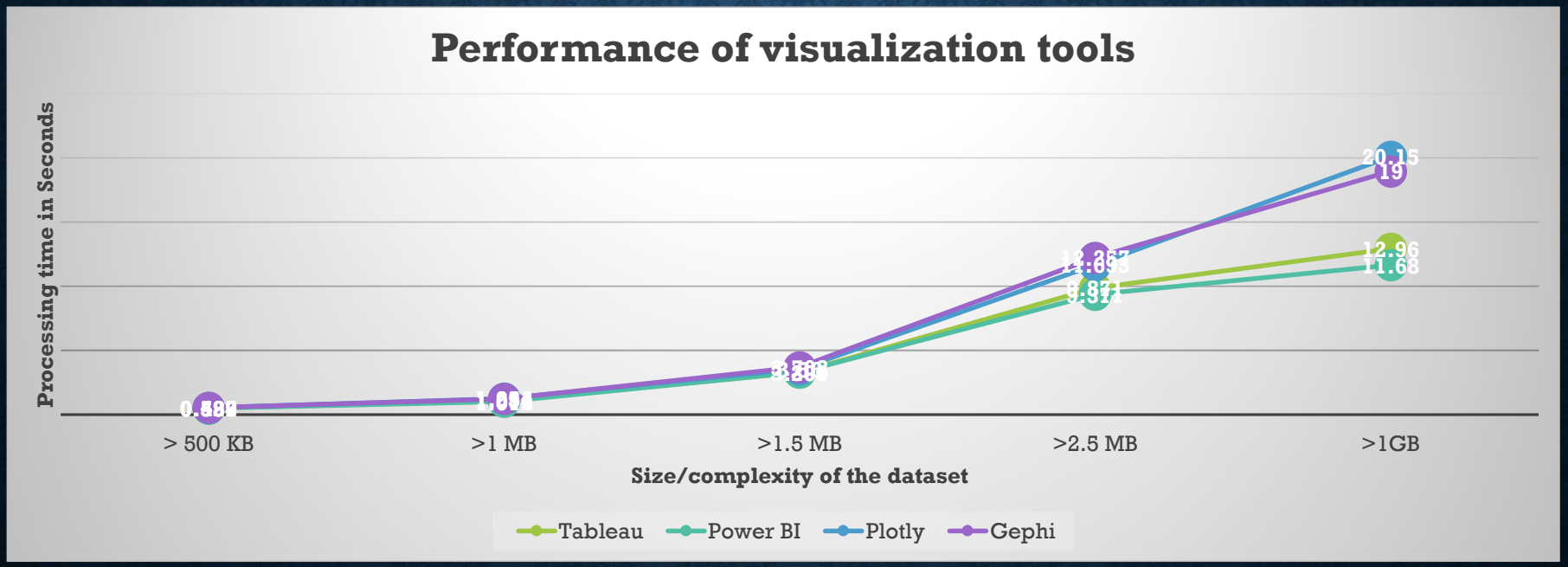
- Graph Size: In Gephi, a primary issue that can lead to an input/output bottleneck is the size of the graph that is being processed. Large graphs may be time-consuming to read and create, slowing down visualization.

- Network delay: Network delay can be a significant impediment when using Gephi to display graphs from a distant server. Long load times and a sluggish visualization might result from slow network connections.

SOLUTIONS TO OVERCOME CHALLENGES IN GEPHI

- Graph Optimization: Reduce the quantity of nodes and edges in the graph to improve it. This can make the graph smaller and boost efficiency.
- Parallel Processing: Implement strategies for parallel processing to hasten the viewing of big graphs. This may include the use of distributed computing or multi-core computers.
- Hardware Optimization: Utilize high-performance drives and optimize disk performance parameters to improve hardware performance. To guarantee that Gephi has adequate resources to process huge graphs, choose a high-performance CPU and RAM.

INTERPRETATION

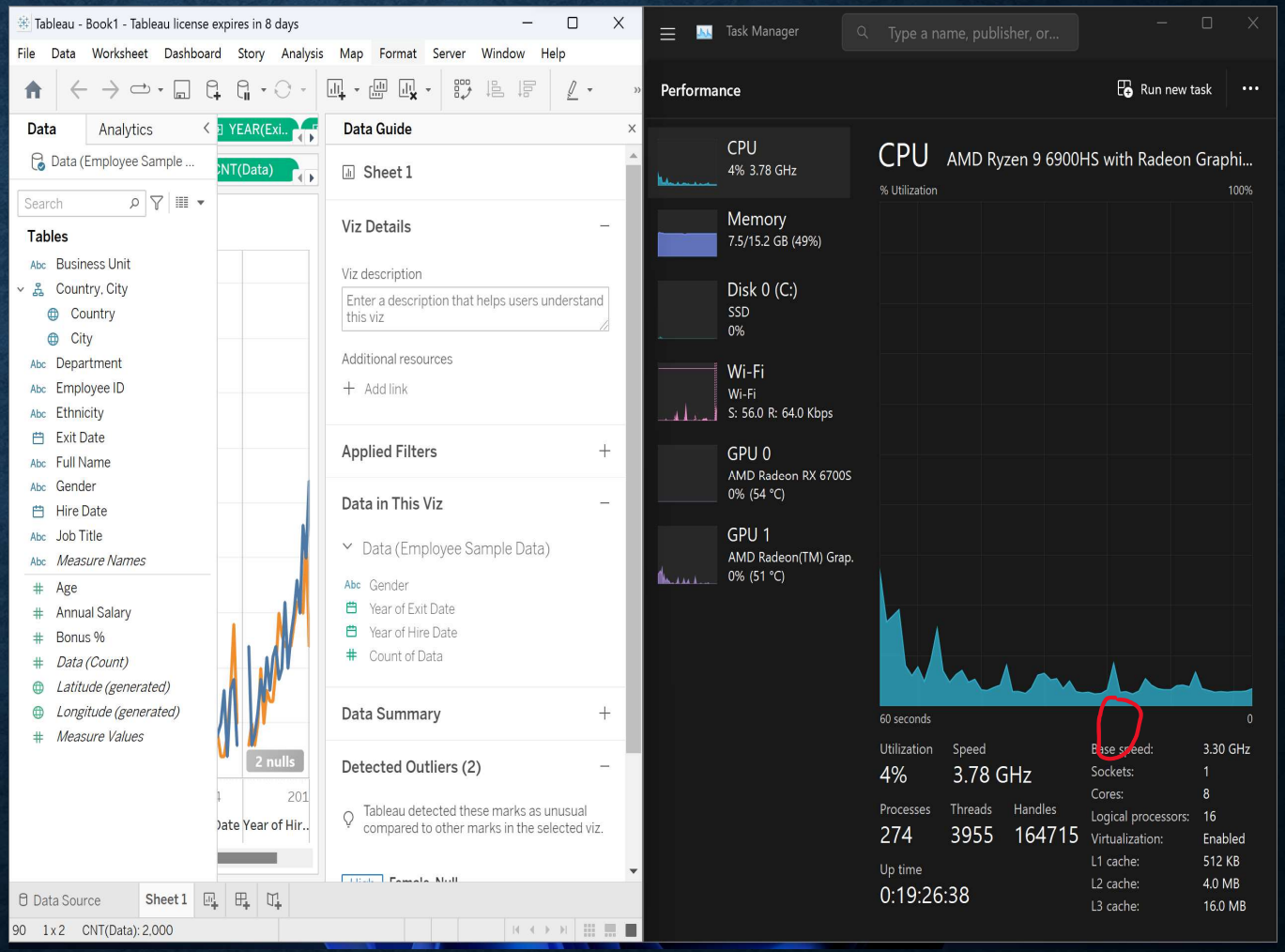


RESULTS

- For the dataset less than 500 Kb, 1Mb, 1.5Mb the results are more similar. That is we cannot observe the difference in real time.
- However, for the dataset which is less than 2.5 Mb we could clearly observe that Power BI was the fastest tool to visualize followed by Tableau, Plotly and Gephi.
- Even though there is a difference in the time taken, Power BI and Tableau are performed very close than the plotly and gephi. The dataset used here was very complex and it was difficult for plotly and gephi due to lack of optimization. Power BI and Tableau are well optimized with respect to parallel processing.
- >1Gb dataset differentiated each tool very well. The Tableau and Power BI automatically splits the data to go on for parallel processing. Plotly and Gephi performed upto extent but could not beat Power BI and Tableau.
- Due to unavailability datasets more complex big data datasets was not interpreted. Only image processing datasets over 1Gb were available for free.
- (Note: The same datasets and same analysis is used for each observation. These tools were tested in windows so the Power BI could exhibit more performance than with other operating systems. However, the difference observed will be negligible to real time as per the theory.)

METHOD USED TO INTERPRET

- Task manager is used to interpret the observation. The x axis is 60 units and while the tools starts to process it gives a spike which can be observed easily in a split screen. So, it was easy to snapshot and calculate the time taken by each tool.
- (Note: There may be some error in the calculation. However, the errors are same for each occurrence there is no variation in observations.)



CONCLUSION

- In conclusion, data analysis and presentation need the use of data visualization technologies such as Tableau, Power BI, Plotly, and Gephi. When working with huge and complicated datasets, these tools must overcome several obstacles, including I/O bottlenecks and parallel processing difficulties. These difficulties may have a substantial effect on the tools' functionality and performance.

- Researchers have suggested several strategies to deal with these problems, including boosting parallel processing capabilities, improving I/O throughput, and creating more effective data visualization methods.

- Furthermore, modern innovations like distributed data storage systems, solid-state drives, and machine learning algorithms can assist boost the efficiency and functionality of data visualization tools.

- The development of more effective I/O methods, parallel processing optimization, and the creation of more complicated data visualization tools that can handle massive and complex datasets in real-time can all be part of Hermes' future work.

FUTURE IMPROVEMENTS

Tableau:

- Adding hybrid cloud support might be a solution to Tableau's input/output bottleneck.
- Implementing AI-driven data preparation might be a further remedy.

Power BI:

- Cloud based caching: Power BI might decrease the quantity of data handled locally and boost speed by caching data in the cloud as opposed to on local devices.
- GPU acceleration: Power BI may handle data more rapidly and effectively, cutting down on the time needed for data analysis and visualization.

Plotly:

- Distributed computing: Plotly could take advantage of the capability of numerous CPUs and GPUs by splitting data and computations over multiple workstations to speed up data processing and visualization.
- Hardware acceleration: Plotly may handle data more rapidly and effectively, cutting down the time needed for data processing and visualization, by using specialist hardware like FPGAs and ASICs.

Gephi:

- Cloud-based computing: Gephi may outsource data processing and visualization to the cloud by using cloud computing resources like AWS and Google Cloud, which would lighten the strain on local PCs and boost speed.
- Graph partitioning: It allows Gephi to more effectively view huge and complicated graphs by taking use of parallel computing.

FUTURE IMPROVEMENTS

Role of Hermes in Tableau:

- Data compression: One approach to reducing the amount of **data exchanged** between Tableau and its data sources is to compress data while it is being transferred. Techniques like delta encoding or dictionary encoding might be used to accomplish this.
- In-memory caching: Implementing in-memory caching to **keep frequently used data** in memory and preventing the requirement for I/O operations.

Role of Hermes in Power BI:

- Columnar data storage: Using columnar data storage, which improves I/O speed by reading only the relevant columns from storage rather than the complete row, is one potential choice.
- Parallel processing: Implementing parallel processing methods, such as multithreading, is an additional option that would permit several I/O operations to take place at once.

FUTURE IMPROVEMENTS

Role of Hermes in Gephi:

- Sample the graph to reduce its size, which might help with I/O operations while displaying the graph. This is one such option. This can include applying methods like importance sampling or random sampling.
- Partitioning the graph into smaller subgraphs that may be handled in parallel is an additional approach that could lessen the I/O load on each individual computer. This can include utilizing strategies like label propagation or partitioning based on modularity.

Role of Hermes in Plotly:

- Data Preprocessing: Pre-processing data to decrease its size before submitting it to Plotly for display might be one possibility. This can involve sampling the data, lowering accuracy, or aggregating the data.
- Distributed computing Utilizing distributed computing techniques to allow data processing to take place across numerous machines would be a different approach that might minimize the I/O load on any one system.


THANK YOU!
Questions?