Project Report on:

Inspecting the main challenges faced in big data analytics tools during visualization:

A Study

Submitted by:

Jaichiddharth Rangasamy Karthigeyan

A20527281

Illinois Institute of Technology

College of Computing

INDEX

Abstract

Data analysis and display technologies such as Power BI, Tableau, Plotly, and Gephi are commonly utilized. However, these tools suffer difficulties due to parallel processing, operating system limitations, and input/output constraints, all of which can have a substantial influence on performance. This paper tries to investigate these issues and provide innovative strategies to solve them in this environment. We suggest employing edge computing and machine learning techniques as potential new solutions for these visualization tools to overcome these difficulties. Edge computing moves computation and data storage closer to the point of use, lowering latency and increasing processing speed. By anticipating which techniques and hardware configurations are most successful for a particular dataset, machine learning algorithms may enhance data processing and visualization tasks. We investigate the issues that these visualization tools encounter, such as the reasons of input/output bottlenecks, and how the recommended solutions might be utilized to overcome them. Furthermore, we provide different solutions for each visualization tool, such as Power BI, Tableau, Plotly, and Gephi.

1. Introduction:

Because of the large volumes of data generated by modern civilization, improved visualization techniques for data analysis and display are required. Visualization tools like Power BI, Tableau, Plotly, and Gephi have become indispensable for corporations, researchers, and people seeking to extract insights from data and make educated decisions.

However, these technologies face major problems due to parallel processing, operating system limitations, and potential input/output bottlenecks. Parallel processing is a computer technology that allows numerous jobs to be processed at the same time, reducing processing time and increasing efficiency. Operating system difficulties, such as memory management and disk input/output, can degrade performance. When the amount of data being processed or sent is too great, input/output bottlenecks emerge, resulting in data transmission delays and performance deterioration.

Existing research has looked into potential remedies for these problems, including data optimization, network optimization, hardware optimization, and parallel processing. To address these issues, nevertheless, new approaches are required as old ones are not always successful. This study intends to investigate the difficulties encountered by parallel processing, operating system problems, and input/output bottlenecks by visualization tools including Power BI, Tableau, Plotly, and Gephi. We also suggest fresh approaches to these problems, such edge computing and machine learning methods.

Edge computing is a distributed computing paradigm that reduces latency and boosts processing speed by moving computation and data storage closer to the point of demand. Machine learning algorithms can accelerate data processing and visualization chores by foretelling the approaches and hardware setups that will work best with a particular dataset. The efficiency of the suggested solutions in enhancing performance and resolving the issues these visualization tools confront are assessed and compared with those of the already available alternatives. The findings of this study can shed light on how visualization tools might be enhanced to function better, and the suggested fixes can serve as a springboard for more study and development in this field.

2. Problem Statement:

The issue addressed in the article is the performance constraints experienced by popular visualization tools like Power BI, Tableau, Plotly, and Gephi due to parallel processing, operating system limitations, and input/output bottlenecks. These issues must be addressed in order to increase the efficiency of these tools in processing enormous datasets and the quality of data analysis and display.

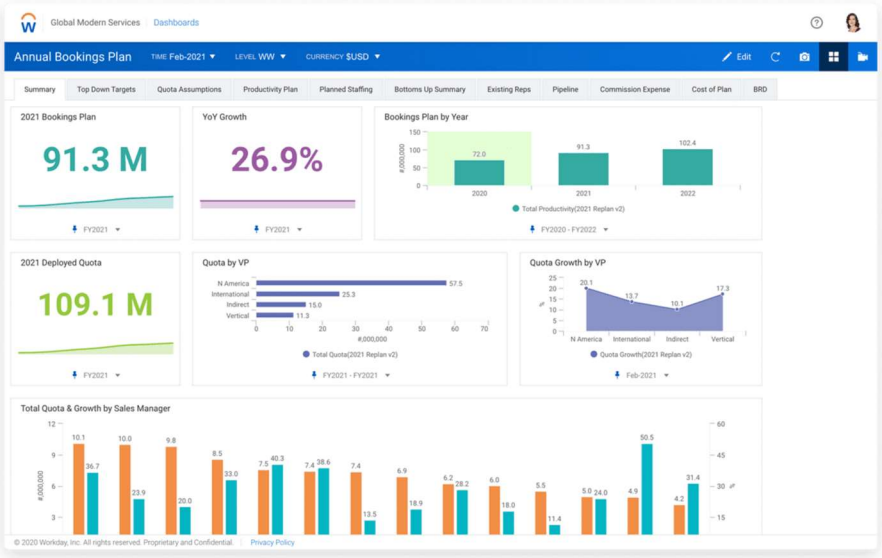
2.1 Problems in existing system

Existing visualization technologies, such as Power BI, Tableau, Plotly, and Gephi, confront several hurdles when dealing with massive datasets and the resulting performance concerns. These difficulties can cause delays in data processing and display, as well as lower productivity and higher expenses.

One of the major issues is connected to parallel processing, or the capacity to conduct numerous tasks at the same time. Parallel processing enables visualization tools to handle enormous datasets efficiently and quickly. Existing systems, on the other hand, may not fully use the available computing capacity, resulting in poorer performance.

Input/output bottlenecks are a major problem as well. Operations known as input/output operations include reading or writing data to and from a storage medium, such as a solid-state drive or hard drive. Data processing and presentation may be delayed as a result of these sluggish activities.

To examine the challenges faced by big data visualization tools such as tableau, Power BI, plot Ly and Gephi. The obstacles related to parallel processing and I/O bottlenecks are studied and suggest solutions to overcome those issues.



3. The study

3.1 Tableau:

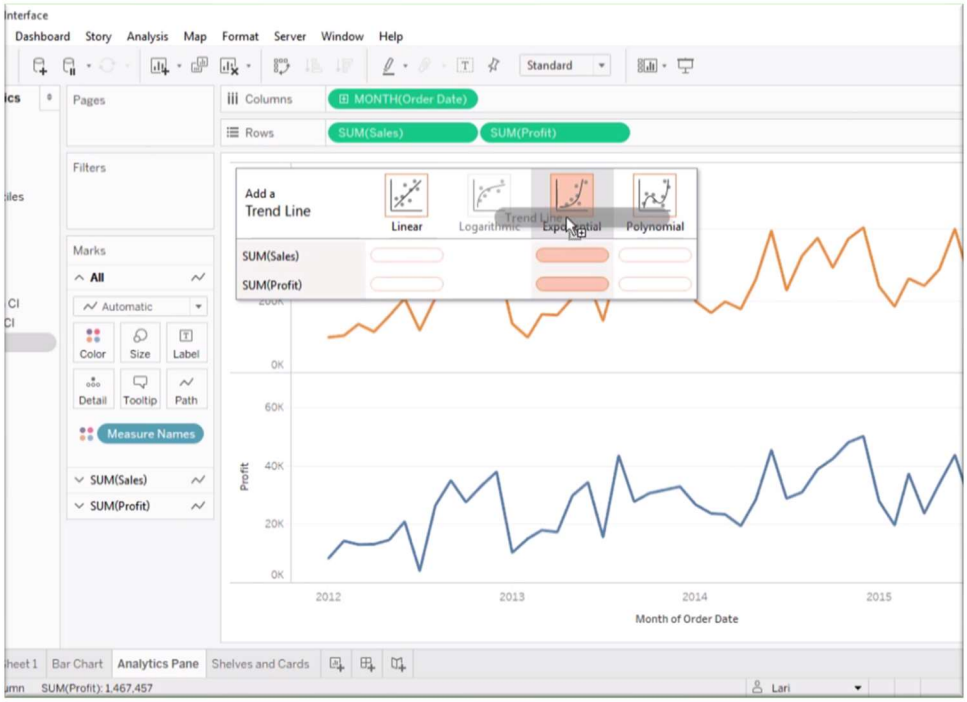


Fig. Tableau Interface

Tableau architecture:

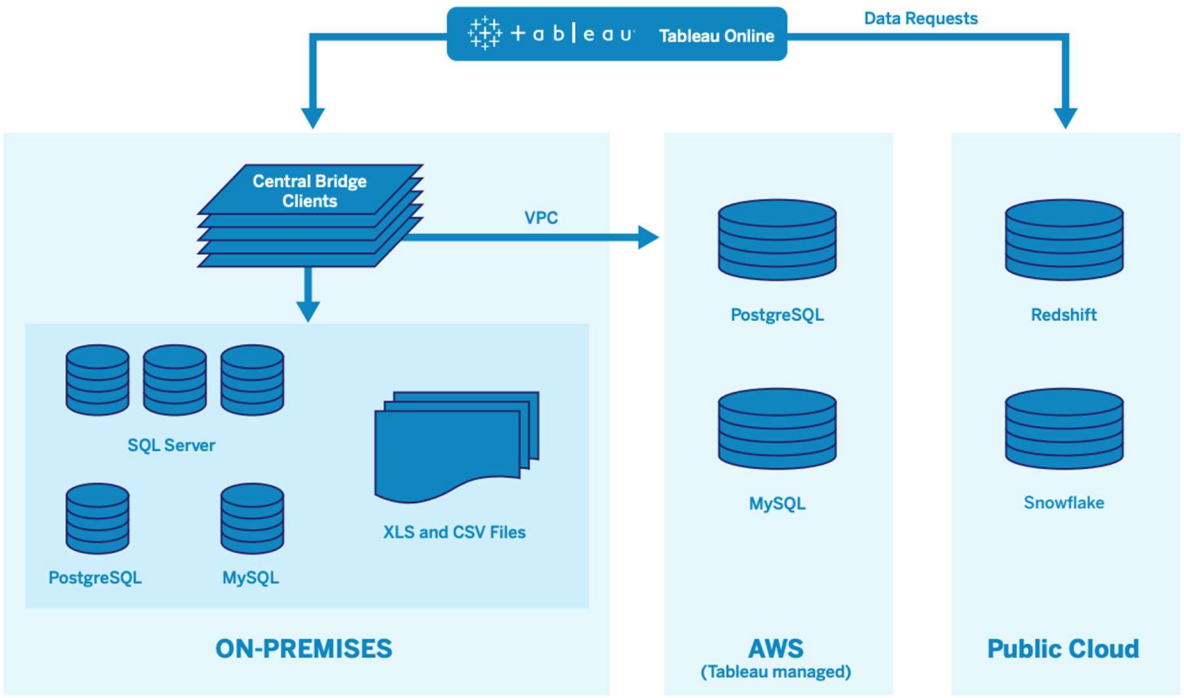


Fig. Tableau Architecture

Users may connect to numerous data sources, convert and analyze data, and create interactive visualizations, dashboards, and reports using Tableau, a potent business intelligence and data visualization application. Tableau's architecture is made up of three key parts:

Data Sources: The Data Sources are the initial part of the Tableau architecture. Spreadsheets, databases, cloud-based data warehouses, and web-based data are just a few of the many data sources that Tableau supports. 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.

Tableau desktop: Tableau Desktop, a self-service analytics and visualization tool used by analysts and data scientists to build and publish interactive dashboards, reports, and visualizations, is the second part of Tableau's architecture. Users of Tableau Desktop have access to a variety of data sources, the ability to mix and blend data from diverse sources, and a drag-and-drop interface for creating visuals.

Tableau server: Tableau Server, a web-based platform that enables users to share, interact with, and distribute dashboards, reports, and visualizations around the enterprise, is the third element of Tableau's architecture. Users may connect to Tableau Server from any device or location to securely view the data and interact with the visuals. Tableau Server also offers governance and security tools to guarantee that the data is safe and complies with all rules and laws of the enterprise.

Parallel processing in Tableau:

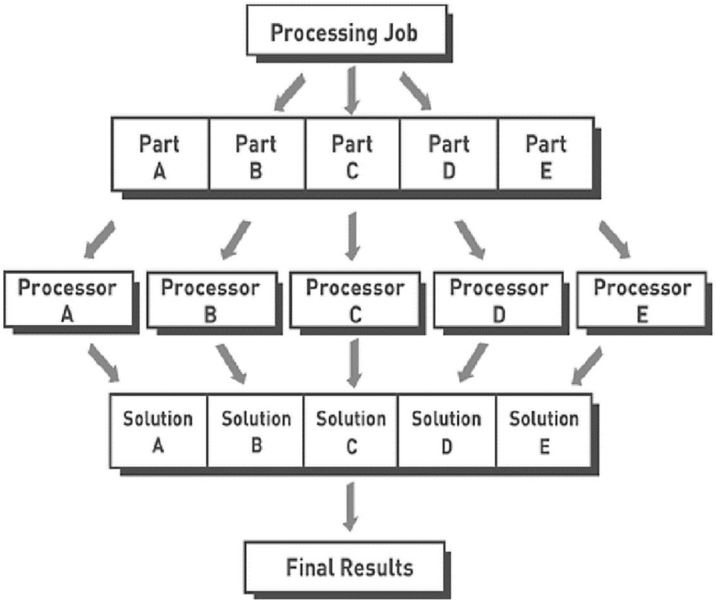


Fig: Parallel processing in Tableau

Due to Tableau's high amount of data production, typical single-threaded processing might become unresponsive and ineffective. Techniques for parallel processing can hasten and improve the visualization process. However, because it calls for abilities and expertise, integrating parallel processing in visualization is a difficult undertaking.

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 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:

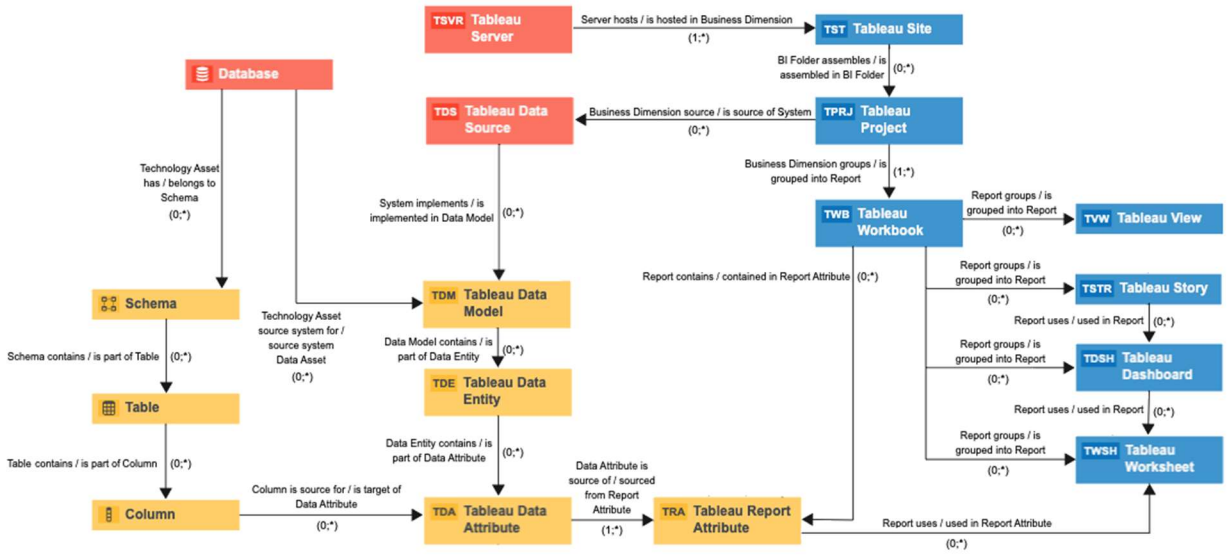


Fig: Tableau desktop

Operating system problems: In order to handle massive datasets, Tableau needs a lot of memory and computing power. Operating system problems with memory allocation, file management, and interoperability with various operating systems are frequently encountered.

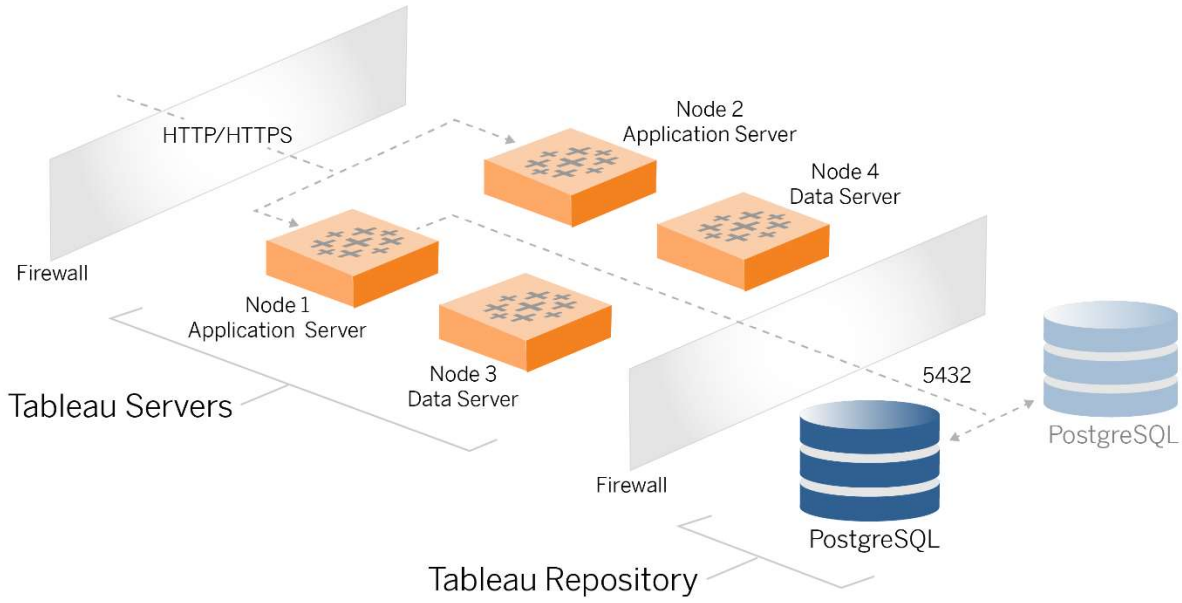


Fig: Tableau server

Tableau offers a component called "Tableau Server," a cloud-based platform that offers specialized computer capabilities for Tableau visualizations, to get around these difficulties. Tableau Server may be utilized with a number of operating systems because it can be installed on Windows or Linux.

Additionally, Tableau provides "Tableau Prep," a data preparation capability that enables data to be cleaned up and modified before being viewed in Tableau. The processing burden on Tableau may be lessened, and visualization performance can be enhanced.

In conclusion, Tableau encounters difficulties during visualization because to parallel processing and operating system limitations. To address these issues, Tableau provides a number of tools and solutions, including Hyper, Data Extracts, Tableau Server, and Tableau Prep. Tableau can manage bigger information and enhance visualization performance by utilizing these technologies, giving its users more precise insights.

Causes of Tableau's input/output bottleneck:

Performance of Data Sources: Tableau can connect to a variety of data sources, including databases, flat files, and cloud-based data storage. 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. Make that the data sources are optimized for the Tableau environment and have adequate capacity to manage the data load.

Reduce the quantity of data that must be exchanged between the data source and Tableau by using data extracts. The performance of data retrieval may be enhanced by using data extracts, which are pre-aggregated subsets of data that can be prepared and optimized for Tableau.

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 high-performance drives and optimize disk performance parameters to improve hardware performance. To guarantee that Tableau has the resources necessary to process huge data quantities, choose a high-performance CPU and RAM.

3.2Power BI:

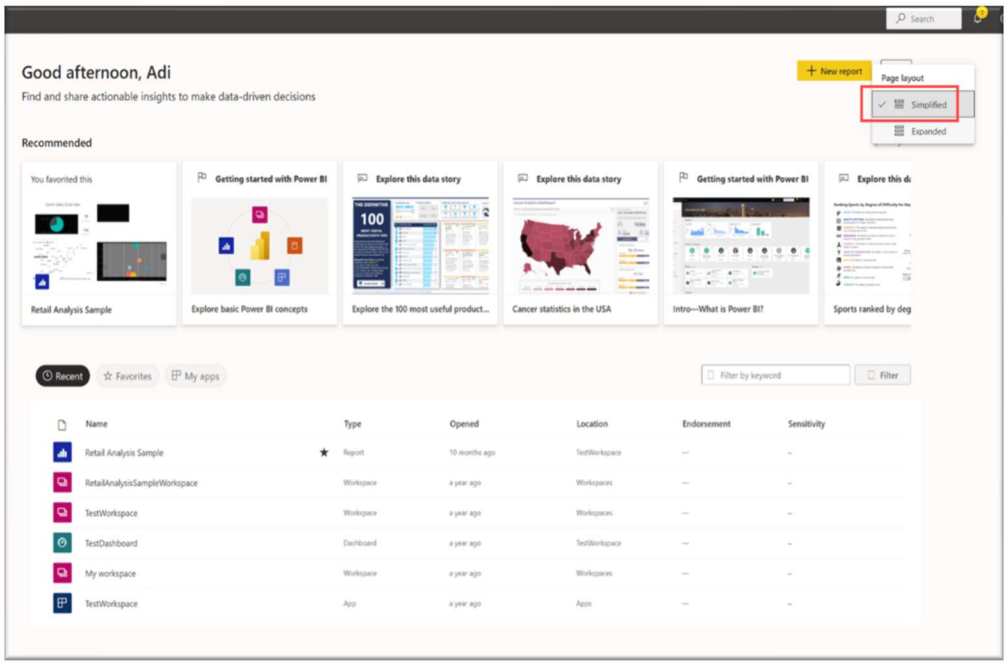


Fig. Power BI interface

Power BI architecture:

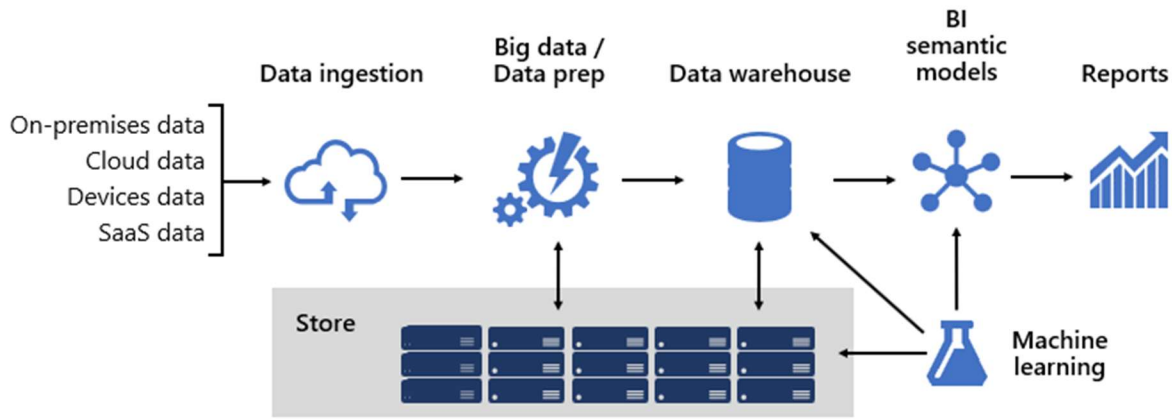


Fig. Power BI Architecture.

Microsoft created the well-known business intelligence product known as Power BI. There are four major parts to its architecture:

Data sources: Data Sources are the initial part of the Power BI architecture. 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. In order to extract data, execute data transformation, and generate data models, Power BI's data engine establishes a connection with the data source.

Power BI Desktop: Power BI Desktop is the second element of Power BI's architecture. To generate data models, infographics, and reports, analysts and data scientists utilize this Windows-based program. 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 Service: Power BI Service is the third element of the Power BI architecture. It is a cloud-based platform used to share and distribute dashboards and reports produced in Power BI Desktop. Power BI Service gives users inside an organization a common area to store, manage, and share Power BI material. Additionally, it offers access restrictions, sharing, and commenting functionality for collaboration.

Power BI Mobile: Power BI Mobile is a mobile app that can be used on iOS, Android, and Windows devices, and it is the fourth element of the Power BI architecture. Users of Power BI Mobile may use their mobile device to view dashboards and reports made in Power BI Desktop and Power BI Service from any place.

Parallel processing in Power BI:

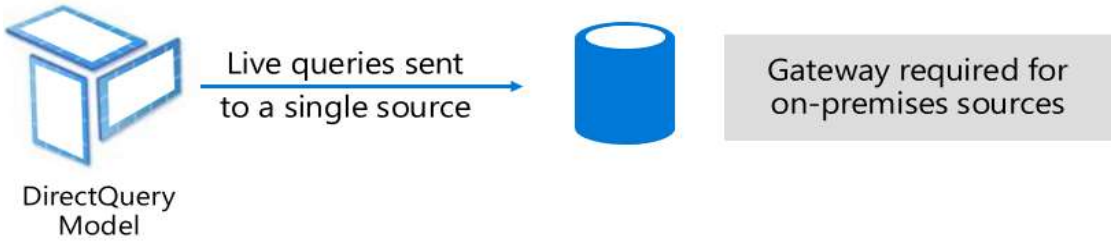


Fig. Parallel processing in Power BI

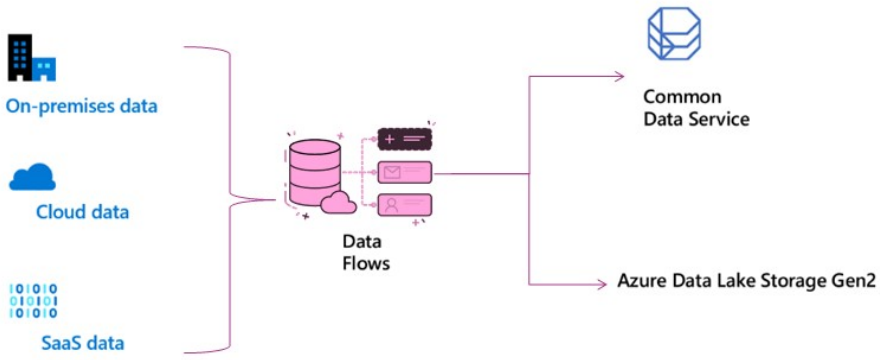


Fig. Parallel processing by Data Flows

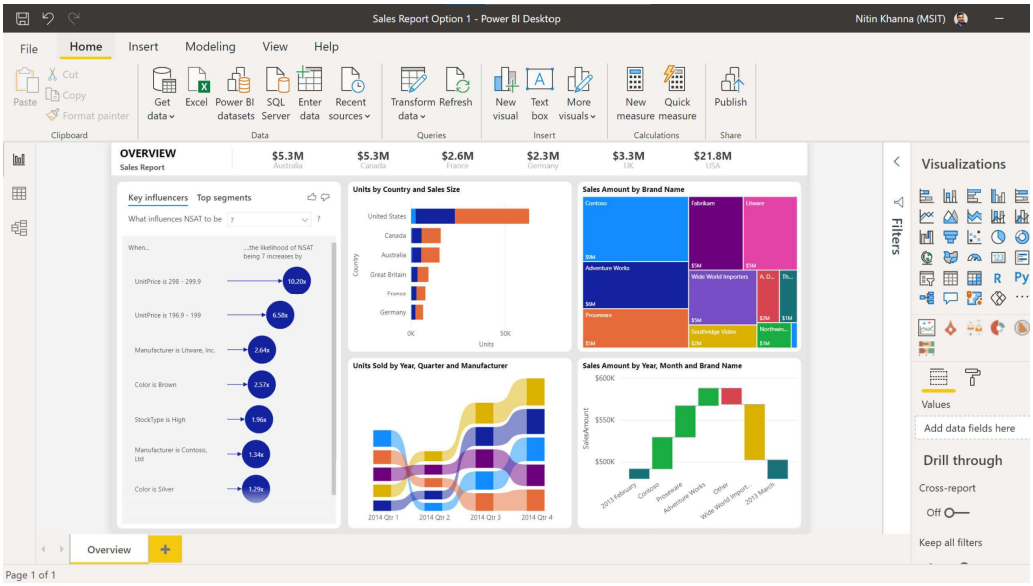


Fig: Power BI interface

Because Power BI creates a lot of data, single-threaded processing can be cumbersome and sluggish. Techniques for parallel processing can hasten and improve the visualization process. However, because it 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," a feature introduced by Power BI, enables data to be queried straight from the source database. Because Direct Query makes use of the processing capacity of the underlying database, Power BI is able 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. By doing this, Power BI's processing burden may be reduced, and visualization performance can be enhanced.

Operating system and Power BI relationship:

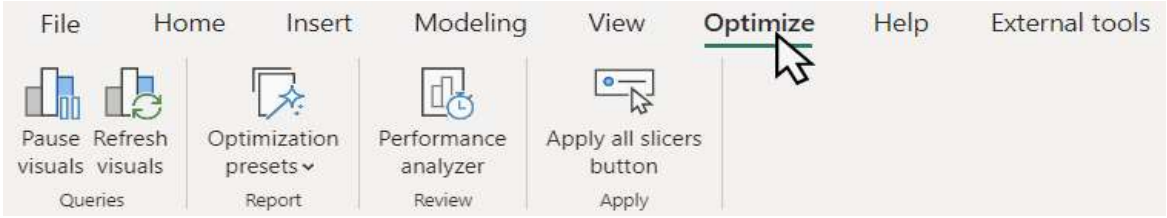


Fig. Optimization in Power BI

To handle huge datasets, Power BI needs a lot of memory and computing power. Operating system problems with memory allocation, file management, and interoperability with various operating systems are frequently encountered.

In order to address these issues, Power BI offers a tool called "Power BI Desktop Optimization," which enhances Power BI Desktop's functionality on Windows 10. This feature provides several improvements, including a 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.

Additionally, Power BI includes cloud-based options, like as "Power BI Premium," which has a dedicated processing and rendering engine for Power BI reports. This can offer a uniform platform for data analysis and visualization while assisting in overcoming operating system problems.

In conclusion, Power BI encounters difficulties during visualization due to parallel processing and operating system limitations. To address these issues, Power BI provides several tools and solutions, including Direct Query, Dataflows, Power BI Desktop Optimization, and Power BI Premium. Power BI can manage bigger datasets and enhance visualization performance by utilizing these technologies, giving its customers access to more precise insights.

Causes of Power BI's input/output bottleneck:

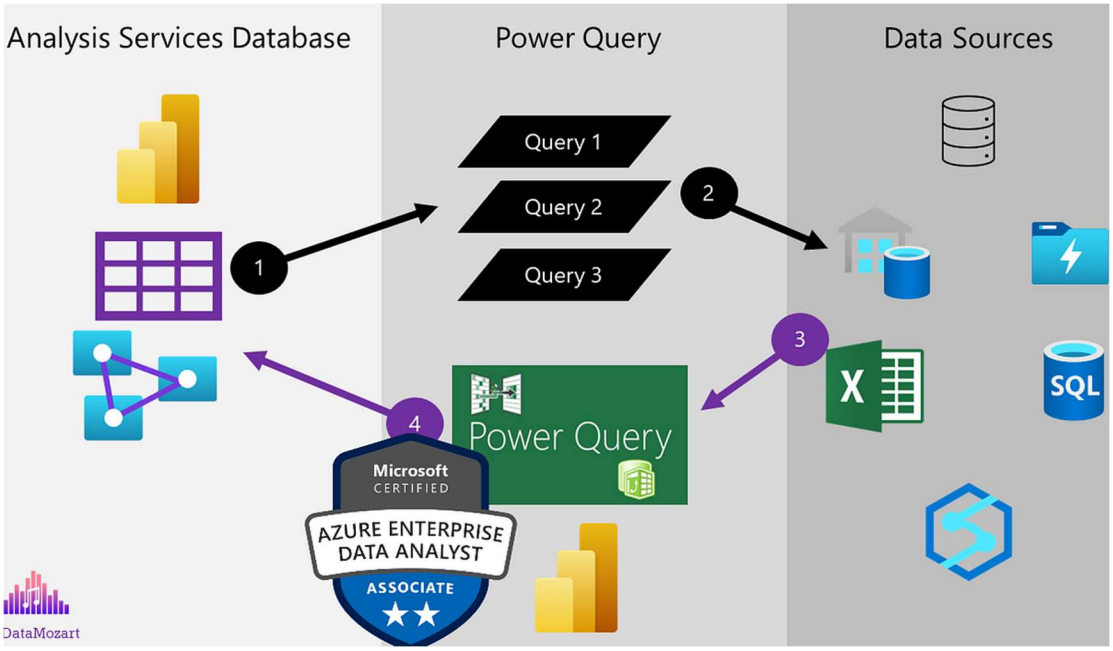


Fig. Power BI bottleneck

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. Make that the data sources are optimized for the Power BI environment and have adequate capacity to manage the data load.
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.

3.3 Plotly:

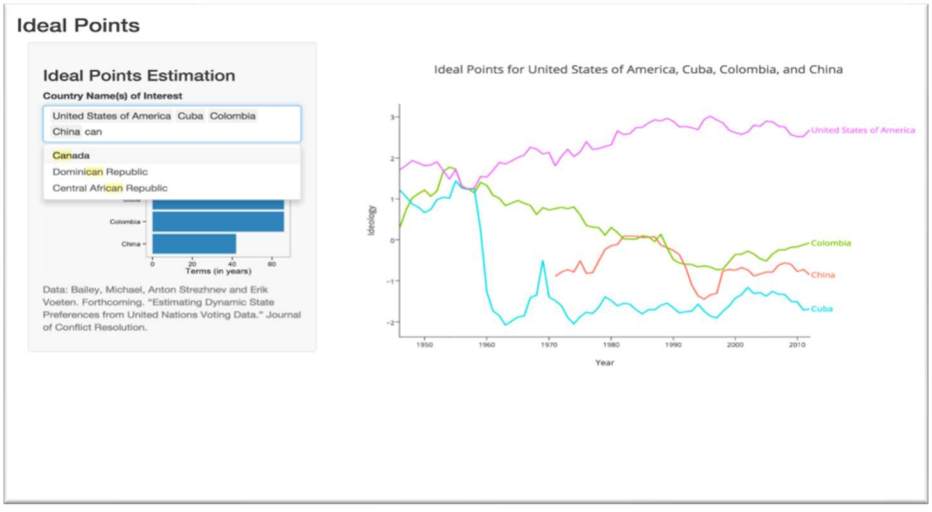


Fig. Plotly Output

Plotly architecture:

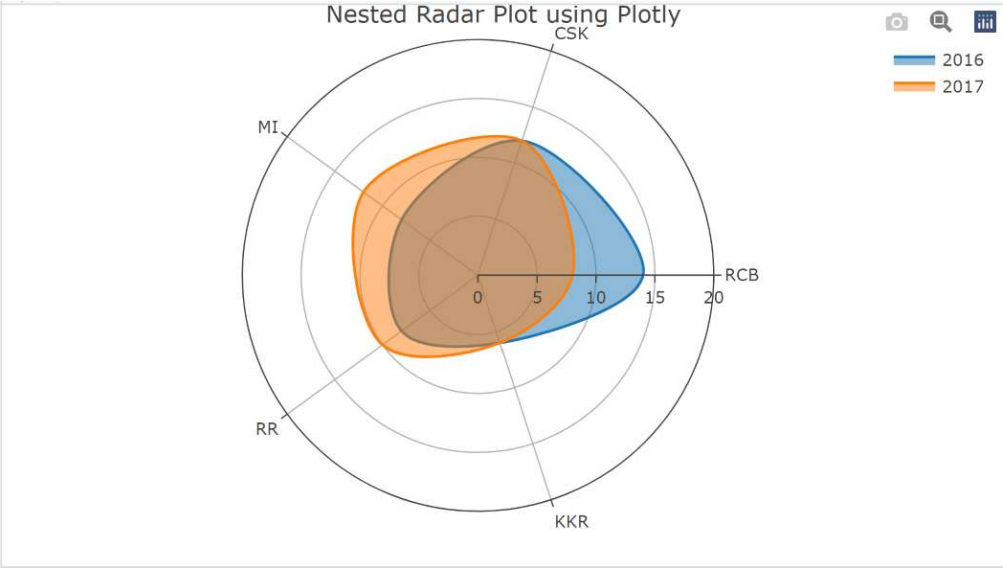


Fig. Nested radar plot in plotly

Users may build interactive charts, graphs, and dashboards using JavaScript, Python, R, and the data visualization tool Plotly. The three primary parts of its architecture are as follows:

Plotly Graphing Library: The Plotly Graphing Library is the initial element of the Plotly architecture. 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: The Plotly Chart Studio is the second element of Plotly's architecture. 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: Plotly Dash is the third element of Plotly's architecture. 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.

In conclusion, Plotly's architecture comprises of Plotly Dash, Plotly Chart Studio, and Plotly Graphing Library. These elements come together to provide data scientists, analysts, and developers across industries a versatile and potent data visualization and dashboarding solution.

Parallel processing in plotly:

Plotly employs parallel processing techniques to speed up the development of charts. Plotly is primarily a Python-based toolkit for building interactive visualizations. When plotting huge datasets, users could encounter performance concerns 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 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:

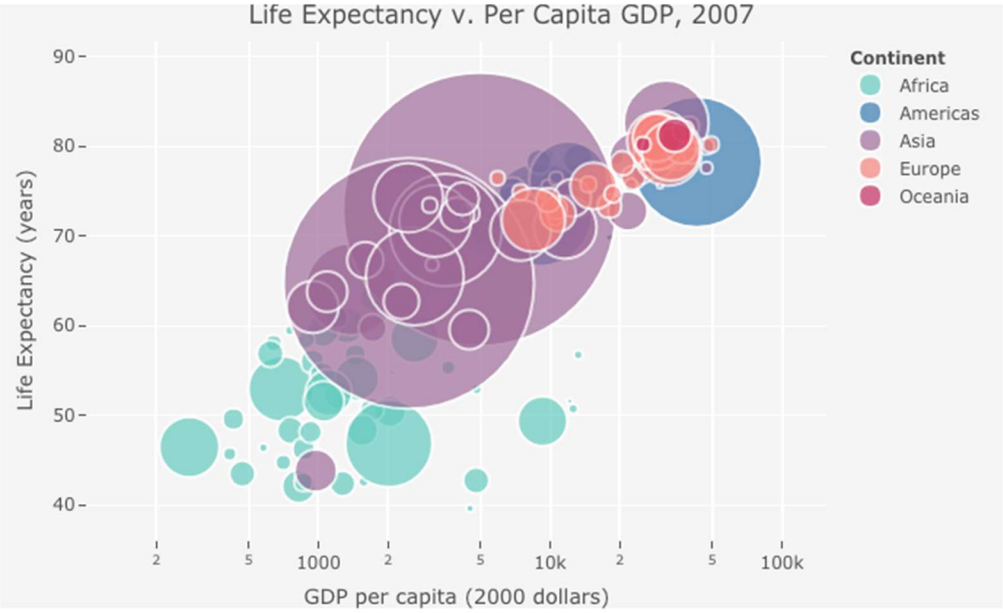
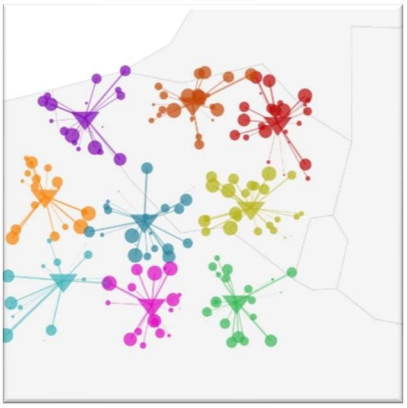


Fig. visualization in plotly

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 visualization.

Network delay: It 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.

3.4 Gephi:

Gephi architecture:

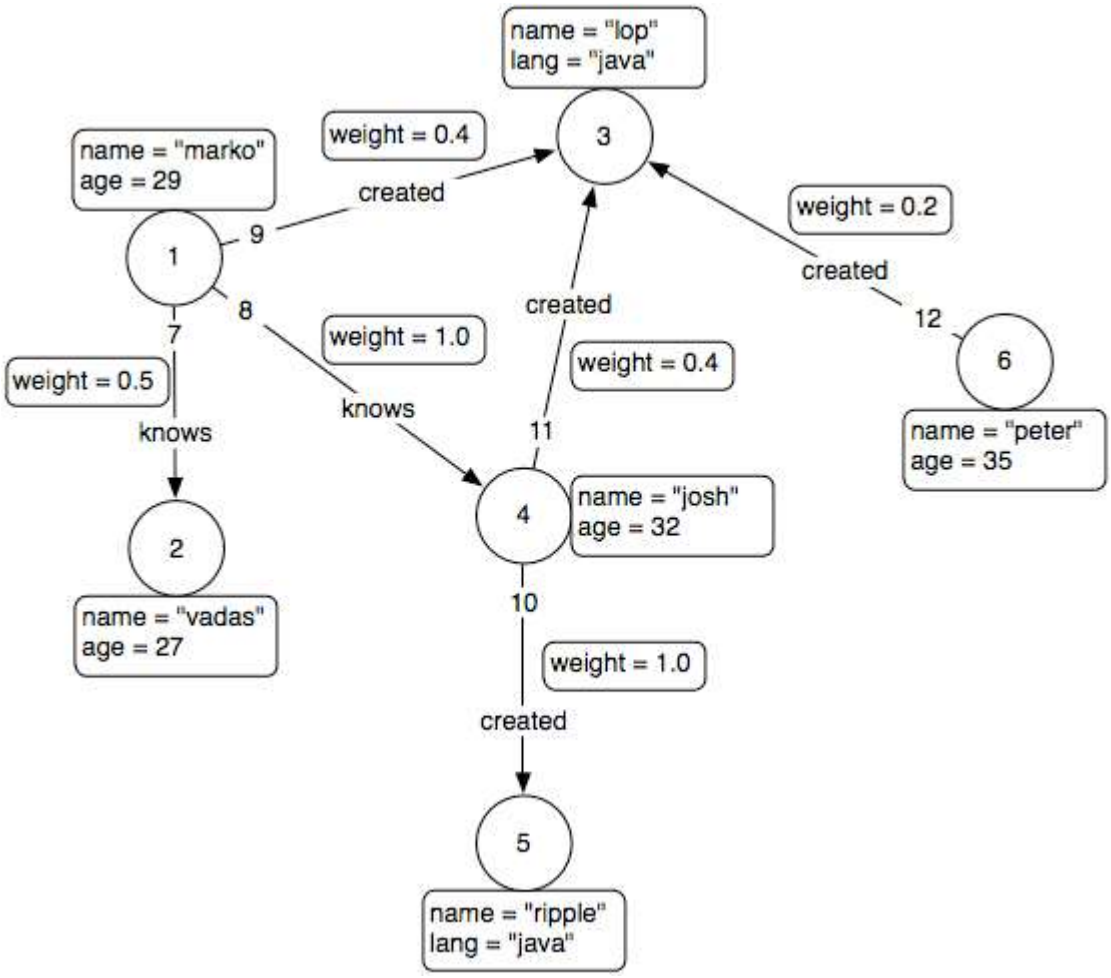


Fig. Gephi working Design.

Popular open-source visualization and exploration software for network research is called Gephi. The three primary parts of its architecture are as follows:

Graph Model: The graph model is the initial element of Gephi's architecture. This is the fundamental data structure that symbolizes the network under study. 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: Gephi Desktop is the second element of the Gephi architecture. A graphical user interface for network viewing and investigation is offered by this desktop program. 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: The Gephi Toolkit is the third element of Gephi's architecture. 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.

The Graph Model, Gephi Desktop, and Gephi Toolkit make up the architecture of Gephi. These elements come together to provide academics, data scientists, and developers working in a variety of industries a versatile and potent network analysis and visualization platform.

Parallel processing in Gephi:

A desktop-based visualization program called 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 concerns.

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 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:

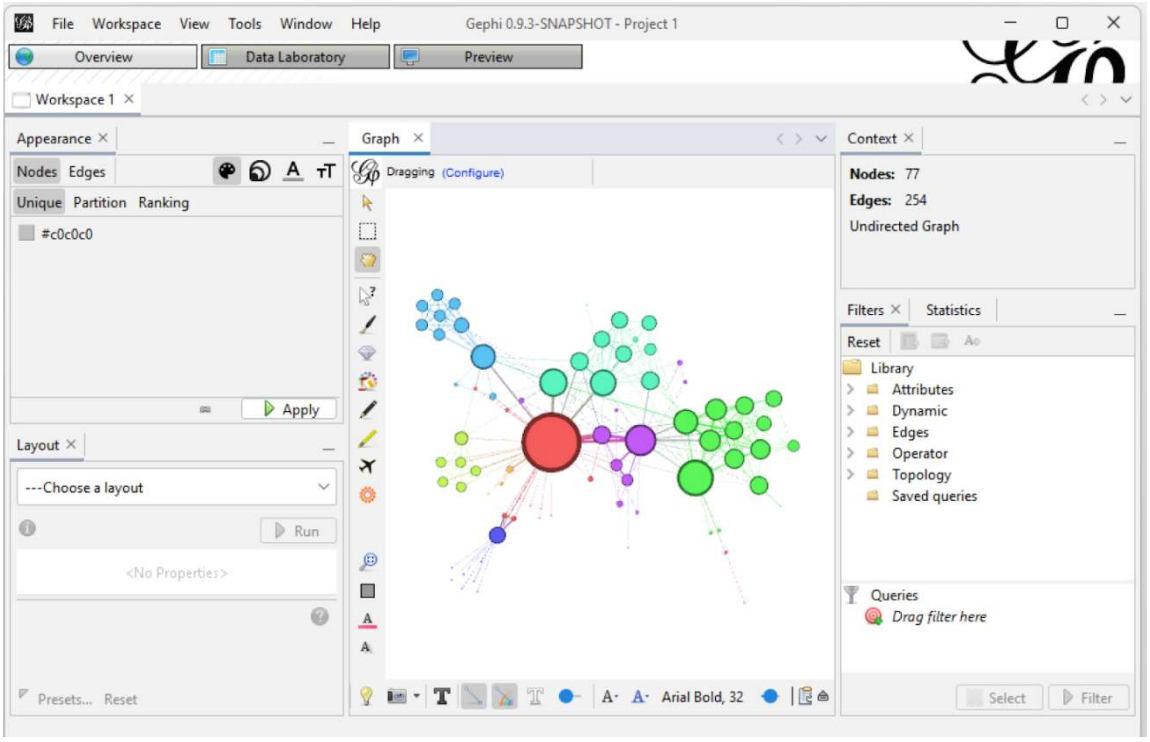


Fig. Gephi user interface

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.

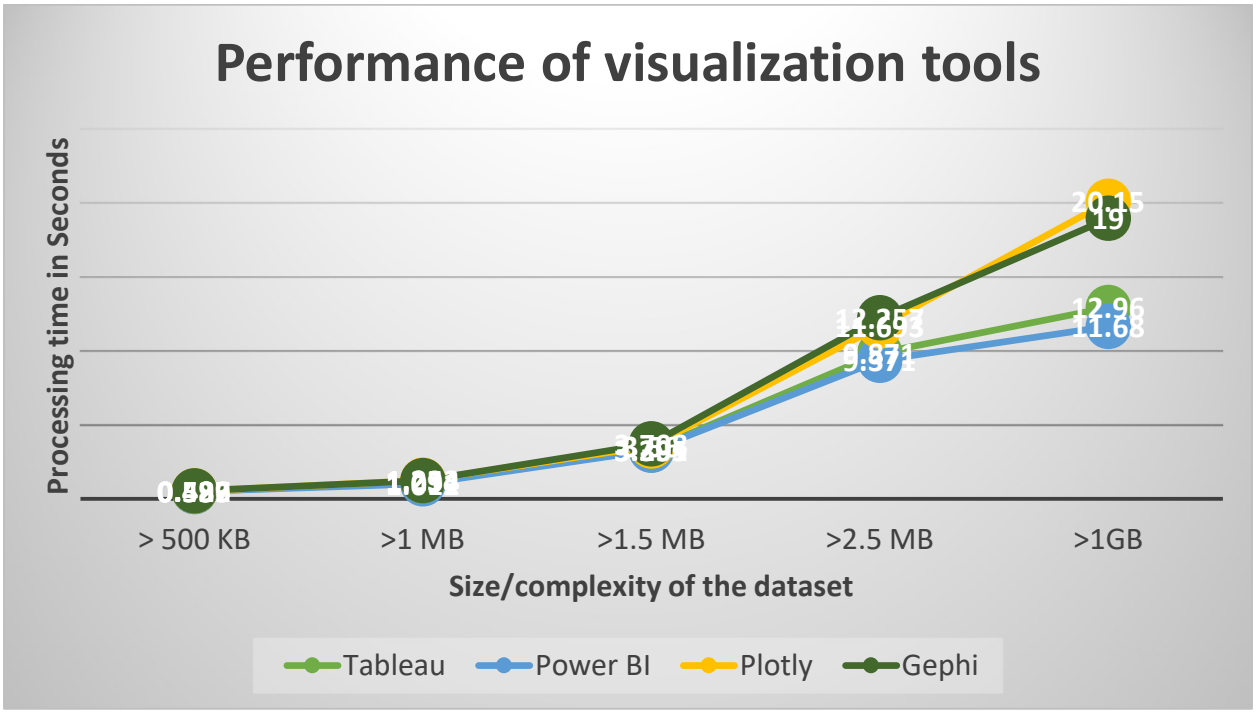
Solutions to overcome challenges in Gephi:

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.

4. 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:

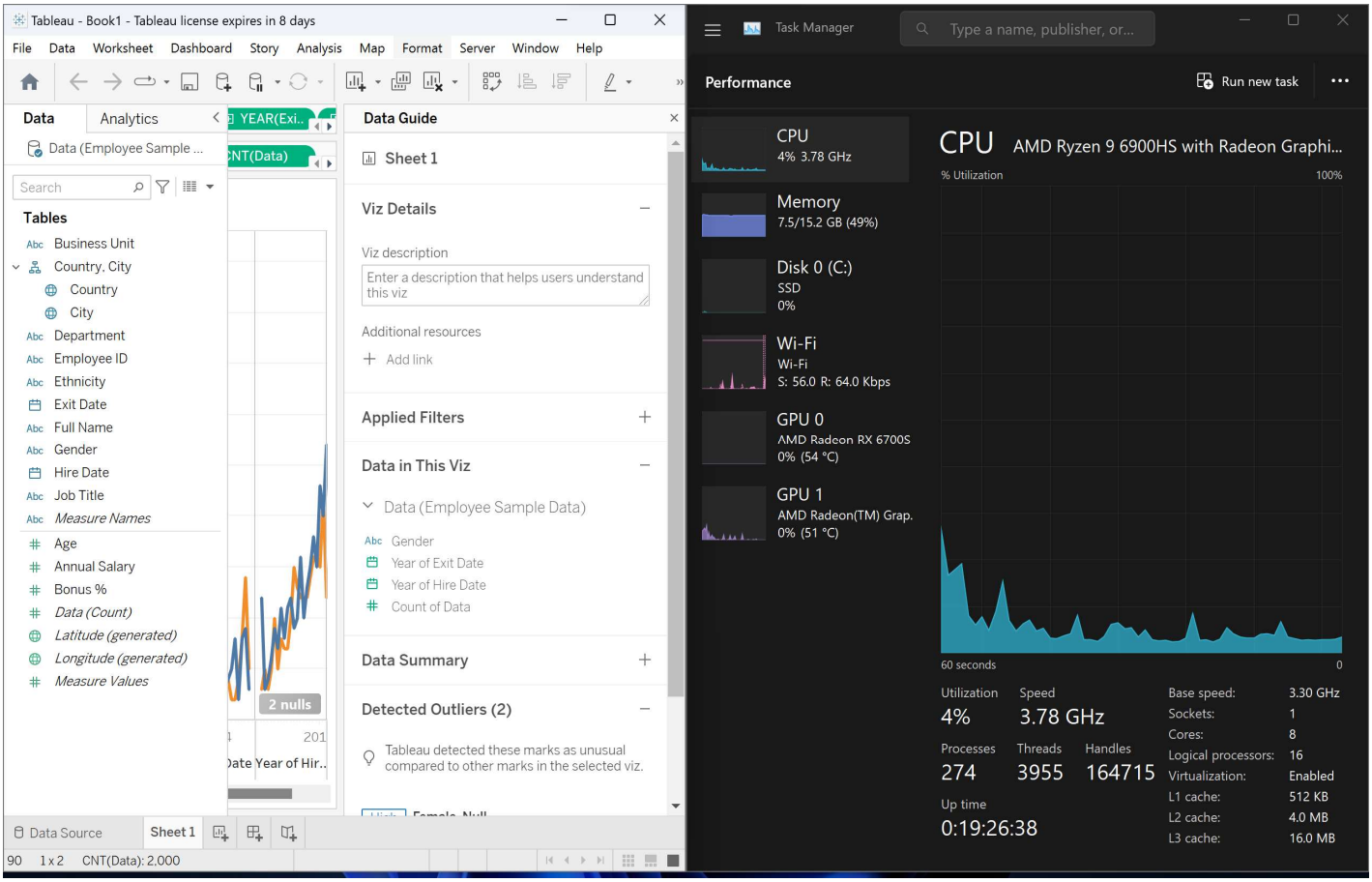


Fig. Task manager

- The Task manager is used to interpret the observation. The x axis is 60 units and while the tools start 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 the same for each occurrence there is no variation in observations.)

5. 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.
Hermes needs to solve the I/O bottleneck to perform better and enable users to manage larger datasets more effectively. 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.

For data science to advance and for users to be able to make better decisions based on huge and complicated information, it will be essential to overcome the issues that data visualization tools confront. Data visualization tools will grow more potent, effective, and user-friendly because of the solutions suggested in this area, opening up a wide variety of applications in several industries.

6. Future Improvements:

4.1 Tableau:

Hybrid clouds Support: Adding hybrid cloud support might be a solution to Tableau's input/output bottleneck. Tableau might decrease the quantity of data handled locally and boost speed by enabling customers to simultaneously store and analyze their data on both local servers and cloud servers.
AI-driven data preparation: Implementing AI-driven data preparation might be a further remedy. Tableau might decrease the time and effort needed for manual data preparation by utilizing machine learning algorithms to automatically clean and convert data, enabling users to produce visuals more quickly and effectively.

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 is another option.

4.2 Power BI:

Cloud based caching: Implementing cloud-based caching might be one solution to the problem with Power BI's input/output bottleneck. 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: Using GPU acceleration for data processing might be an additional alternative. capability BI may handle data more rapidly and effectively, cutting down on the time needed for data analysis and visualization, by making use of the parallel processing capability of GPUs.

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

4.3 Plotly:

Distributed computing: Implementing distributed computing might be a solution to Plotly's problems with parallel processing. Plotly could take advantage of the capability of numerous CPUs and GPUs by splitting data and computations over multiple workstations. This sped up data processing and visualization.
Hardware acceleration: Implementing hardware acceleration for data processing might serve as an additional remedy. Plotly may handle data more rapidly and effectively, cutting down on the time needed for data processing and visualization, by using specialist hardware like FPGAs and ASICs.

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.

4.4 Gephi:

Cloud-based computing: Implementing cloud-based computing might be a viable answer to the problems with parallel processing in Gephi. 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: Implementing graph partitioning techniques to split big graphs into smaller sub-graphs that may be handled in parallel is an additional option. This method of graph division allowed Gephi to view huge and complicated graphs more effectively by taking use of parallel computing.

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.

7. References:

[1] Ali, Syed Mohd, Noopur Gupta, Gopal Krishna Nayak, and Rakesh Kumar Lenka. "Big data visualization: Tools and challenges." In *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)*, pp. 656-660. IEEE, 2016.
[2] Hajirahimova, M., and M. Ismayilova. "Big data visualization: Existing approaches and problems." *Problems of Information Technology* 9, no. 1 (2018): 65-74.
[3] Gahi, Youssef, Mouhcine Guennoun, and Hussein T. Mouftah. "Big data analytics: Security and privacy challenges." In *2016 IEEE Symposium on Computers and Communication (ISCC)*, pp. 952-957. IEEE, 2016.
[4] Raghav, Raja Surya, Sujatha Pothula, T. Vengattaraman, and Dhavachelvan Ponnurangam. "A survey of data visualization tools for analyzing large volume of data in big data platform." In *2016 International Conference on Communication and Electronics Systems (ICCES)*, pp. 1-6. IEEE, 2016.
[5] Caldarola, Enrico G., and Antonio M. Rinaldi. "Big data visualization tools: a survey." *Research Gate* (2017).
[6] Otero, Carlos E., and Adrian Peter. "Research directions for engineering big data analytics software." *IEEE Intelligent Systems* 30, no. 1 (2014): 13-19.
[7] Employee attrition Dataset: https://www.kaggle.com/datasets/pavansubhasht/ibm-hr-analytics-attrition-dataset
[8] Video game sales dataset: https://www.kaggle.com/datasets/gregorut/videogamesales
[9] Plotly for R: https://plotly.com/r/getting-started/
[10] Gephi: https://github.com/gephi/gephi/releases/download/v0.10.1/gephi-0.10.1-windows-x64.exe
[11] Power BI: https://aka.ms/pbidesktopstore
[12] Tableau: https://www.tableau.com/products/desktop