CSE 579 - Knowledge Representation

Automated Warehouse Scenario:

The project successfully implemented a warehouse management system using Answer Set Programming (ASP). This system optimizes order fulfillment by employing robots to retrieve items from shelves and deliver them to designated picking stations within a grid-based warehouse setup. The approach involved declaratively modeling the entire scenario in ASP, specifying the warehouse elements (robots, shelves, stations) along with their attributes and governing rules. Actions such as robot movement and shelf handling were represented as ASP choice rules. Crucial hard constraints were imposed, such as preventing robot collisions, ensuring shelves are not dropped on highways, and enforcing validity conditions. The ASP model, in conjunction with specific problem instances, was processed by the Clingo ASP solver to generate stable models that outline conflict-free plans for robots to efficiently complete orders. Visualization of these plans was facilitated by the ASPrilo tool. Through iterative refinement and testing on provided scenarios, the project showcased ASP's efficacy in tackling convoluted combinatorial problems with multiple constraints, underscoring its relevance in warehouse automation and supply chain management. The project employed the Clingo language for ASP modeling and problem resolution.

CSE 578 - Data Visualization

2020 VAST Challenge MC-1: Graph Analysis

The project tackled the challenge by developing an interactive web-based dashboard for analyzing and pinpointing the potential culprit group behind a hypothetical global internet outage. The dataset provided included a base communication network graph depicting suspicious activities, along with five alternate graphs for comparison, supplemented with demographic, spatial, and temporal data. Leveraging D3.js, the solution crafted interconnected visualizations such as network diagrams, bar charts, maps, and temporal plots like stream graphs and heatmaps to juxtapose the template network against the five alternatives. Innovative features like hover-triggered donut charts on nodes and interactive filters were incorporated. Employing the Five Design Sheets methodology ensured thoughtful design for effective visual representation, layout, and interactivity. Through meticulous analysis spanning various visualizations, including comparison of network structures, edge strengths, and temporal/demographic patterns, the solution identified the candidate networks most closely resembling the template, showcasing proficient data visualization techniques for investigative tasks.

Automated Warehouse Scenario

Yukta Sarode
ysarode@asu.edu

***Abstract*—To cater to the ever growing needs of demand and supply, big companies like Amazon, Walmart, Ikea, Target are adopting automation. One such use case is automation of warehouses. In order to fulfill 'one-day delivery' warehouses can deploy robots to deliver products to picking stations for fulfilling orders. This solution can improve efficiency and accuracy of order fulfillment, reduce manpower and help stay companies ahead of the supply chain logistics and its competitors. Hence, this paper proposes a solution for the automated warehouse management problem.**

***Index Terms*—Answer Set Programming (ASP), choice rules, First-order Logic (FOL),Knowledge Representation & Reasoning (KRR),**

I. Introduction

The given problem of "Automated warehouse Scenario" is a simplified version of automated planning done in Amazon warehouses which was also included in the ASP Challenge 2019.

The project aims to develop a planning system which is automated for a warehouse that consists of robots delivering products to picking stations to fulfill orders. The final goal is to fulfill all orders efficiently and accurately, where time efficiency can be counted in steps and each robot may perform one action per time step.

The products are kept on the shelves and the robot has to move the shelf to its proper pick up station. The warehouse is in the form of a rectangular grid like structure which consists of highways where the robot can move. Each robot can only have a unique location and can move between the two adjacent cells of the grid in only two ways:

- Horizontal
- Verticle

The design of the robot is as follows:

- Flat
- Can move under the shelves and pick them up
- Cannot move under a shelf while carrying one.

The robot can do the following actions:

- Pick up
- Put down
- Deliver
- Stay idle

Products can be placed on shelves in the following ways:

- Position of each product should be unique
- Different shelves can have products of the same type

Some more constraints for the robots include:

- Preventing collision between the robots
- Ensure robots do not put down shelves on the highways

The challenge is to complete all requests efficiently by defining hard constraints such that while keeping in mind that the scenario is always changing (dynamic) and the condition of each item might change over time.

II. Description of Solution

Due to COVID-19 there was a boom in the e-commerce industry. Due to the sales and reduced prices on the platform people started using them even more. To handle such demand and supply huge warehouses came into picture for efficacious inventory management. Resources like time, money and energy need to be optimized. Since the warehouses are not completely automated, the solution proposed by me will help the retailers optimize resources.

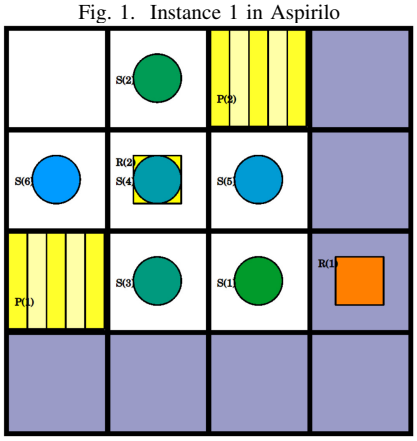
With the help of lectures taught in the class the design and implementation of the functionality for the warehouse and robot was formulated using Clingo (ASP Tool). The system will be automated based on the following things:

- object declaration
- object sorting
- independent domain axioms
- exogenous fluents
- existence and uniqueness of fluents
- inertial fluents

For finding the solution, I adopted the step by step method to formulate the problem statement since it is fairly complex. After watching the project video and understanding the description of the problem. I started with splitting the problem in smaller easy problems. So initially, I created all the possible paths/actions a robot can perform by outlining how the system will work on paper. I did this based on the initialization files given to us to get a rough idea. Later I decided to find the conditions for constraints for the various actions to generate a stable model.

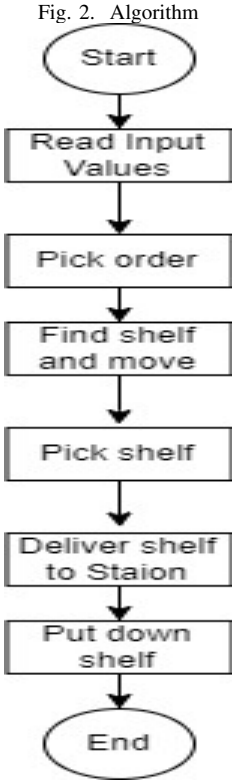
The next step was to take a look at the initialization files to get to know the input. I got the basic warehouse scene data after translating the input. Later, I brainstormed to convert it into an easier format for implementation purposes, for this I also used the visualization software ASPIRILO. In Fig 1, I have visualized instance 1. Initially we can see a 4X4 grid having a purple color highway, the circles(S) denote shelves, the square(R) boxes denote robots and the striped grid(P) boxes denote picking stations. The shelves have the products. This helped me to assert the items in the project. I identified those to be the shelves, robot, picking station, highway, nodes, number of rows and columns. I also understood that to track the count of the robots I need to take care of the concurrent actions performed, this would also be beneficial in identifying

Fig. 1. Instance 1 in Aspirilo



if the robot is inside the grid or not. In this way I understood all the materials which were provided to me and decided to device a simple algorithm (Fig 2) for it so that I can track how the robots are performing.

Fig. 2. Algorithm



The next crucial step was implementing the constraints, since it was very challenging and it quite often reached unsatisfiability if the constraints were not written efficiently. So the process was to understand how the robot will move(behavior), then break the difficult constraint into smaller rules, write those rules in ASP and lastly run them in CLINGO. So I started with a free world which had no constraints.

To simplify the problem formulation, I designated a specific node as the objective node and managed the movement of the robots within the provided framework. As it progressed, I introduced more limitations and actions for other elements in the system. To achieve the best/optimal outcome and for the purpose of troubleshooting, I began programming the 'pickup' and 'putdown' actions of the robots. The ultimate goal was to reach a state where a robot had a shelf on top of it. After a lot of debugging I was able to accommodate all the constraints in CLINGO. I noticed there were a lot of overlaps when i defined the action and move rules when it allowed for only one move:

```
moveAction(R,move(DX,DY),T):move(DX,DY)1:-
R=1..RC, robotsCount(RC), T=0..n-1
```

Here the problem was that I had overlooked the process of grounding it for the move(XX,YY) part, which in duplicates at Time 0. I remembered in the lectures it was mentioned that we should make use of the ground when we are thinking of the rule and skip it when it is present in the body. So when I took care of this non-unique actions were not found.

I also found that it was necessary to determine how many robots are present in a specific situation and also other factors like the number of rows and columns of the grid. So for this the use of the aggregation function Count was useful.

```
numRobots(NR):-
NR=countR:init(obj(robot,R),
value(at,pair(X,Y))).
```

Once the individual rules were established, the next step was to integrate them into a comprehensive rule that would ensure the desired outcome without any interference. This allowed me to assemble the functionality of a robot that could move to nearby cells, collect shelves, transport them to appropriate stations, avoid collisions, and perform other essential tasks.

III. Results

Based on the instances provided, I created a 4x4 grid. It contained highways running along the periphery of the grid. The grid has the robot, product,order, pickup stations and shelves. With the help of testing, I have tried to minimize the time taken by robots to complete an order. And also minimized the no.of actions taken by robots using the instance files. In Fig 3, we can see the output for instance 1, the model obtained was 1 with optimization of 64. With the help of ASPIRILO, I visualized the initial position of all the entities of the grid. The robots do not have any shelves on them initially and the location of the robot or shelf is unique. The output of Instance 2(Fig 4), 3(Fig 5), and 4(Fig 6) are given below in detail.

In Table 1, I have stated the results obtained. Some analysis on the process is as follows:

- Managed to follow all the constraints mentioned in the problem statement by surveying the outputs of five instance files.
- Managed to avoid collision after satisfying all instances with optimization in terms of actions and time even after increasing the no.of robots or shelves. All robots reached the end goal state.

Fig. 3. Instance 1 in Clingo

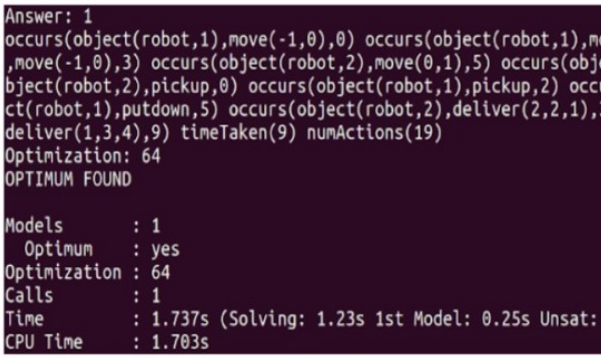
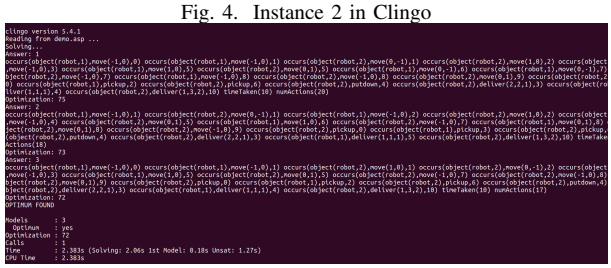


Fig. 4. Instance 2 in Clingo



- Ensured that the robots stay in the grid and took one step at a time.
- I saw that even the shelves were not dropped at highways, ensuring every item is placed on the proper picking station.
- Some random movement was seen after increasing the robots as a result minor updation was required which required a lot of troubleshooting and debugging.
- Due to the constraint where robots are only allowed to carry one shelf, the time taken by robots was directly proportional to the number of orders. However, I also observed that more robots means faster delivery due to simultaneous movement.
- Eventually all instances were satisfied with no collision between the robots.

Based on the above results, I can conclude that:

- The use of automated robots for product handling has potential applications across diverse industries, including restaurant food delivery and one-day delivery services.
- Answer Set Programming (ASP) and Knowledge Representation and Reasoning (KRR) are powerful tools for solving complex real-world problems in various domains, including Computational Biology, Workforce Management, Intelligent Call Routing Protocols, and Decision Support Centers.
- ASP's various features, such as optimization statements, recursive definitions, default negations, aggregates, external atoms, and weight constraints, make it useful for solving a variety of problems, from warehouse management and supply chain optimization to drug discovery and employee scheduling.

TABLE I
RESULTS

Instance	Step	Optimization	Model	Time
1	10	64	1	1.434 Sec
2	12	72	7	3.991 Sec
3	7	31	2	0.219 Sec
4	5	20	1	0.125 Sec
5	7	31	2	0.303 Sec

Fig. 5. Instance 3 in Clingo

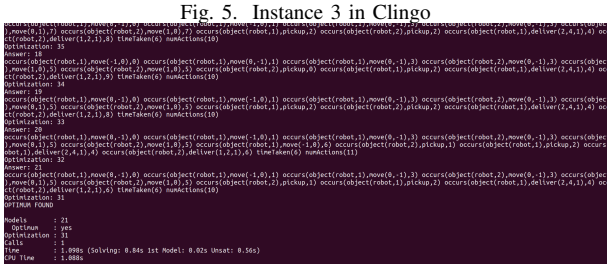
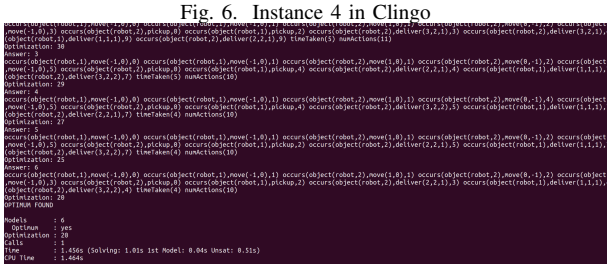


Fig. 6. Instance 4 in Clingo



IV. LESSONS LEARNED

Answer Set Programming (ASP) was thoroughly covered in CSE 579 as a method of problem-solving. I was better able to comprehend how transition systems function, how to design actions and constraints, and other concepts thanks to examples like the blocks world and monkey banana challenges. I discovered that ASP is a potent programming language that enables the application of strict constraints. My understanding of the entire ASP problem-solving process improved as a result of implementing constraints and troubleshooting problems.

One of the most important skill that I gained during this project is understanding the code given and formulating new code according to it. Since the project required us to apply a lot of propositional logic, I am now able to understand the complex concepts related to ASP. Due to the problem statement being a real-world scenario I got an idea of where ASP can be the most optimal language to be used. After using brute-force technique I realized that I could form the constraints smartly if I take into consideration the behavior of the robot.

I successfully managed to solve all the given instances with the least no.of steps in an optimal way. With the step by step implementation, I was able to divide the code into Fluents, Goals, Law of Inertia, Constraints and Action(pickup, deliver,move). Debugging missing restrictions and using the ASPRILO visualizer helped me understand the process better.

The project showed me the importance of looking for edge cases and their impact on the solution. The robots were able to complete all orders with 0 collision in the least amount of time. The learning curve was quite steep during the project but I have gained valuable insights related to this course due to this project. Overall, I learned from the project how Knowledge Representation and Reasoning (KRR) and Artificial Intelligence (AI) can be used to tackle challenging issues.

REFERENCES

[1] Potassco - clingo documentation *https://potassco.org/clingo/*
[2] Joohyung Lee, CSE 579 Lecture Videos *Arizona State University*
[3] Potassco - asprilo documentation *https://potassco.org/asprilo/*

CSE 578 Personal Reflection Report

Yukta Sarode
Arizona State University
Tempe, Arizona
1225266406
ysarode@asu.edu

I. Introduction

The primary goal of the project is to address the VAST MC - 1 Challenge (2020) [1], which presents a hypothetical scenario of a global internet outage. The objective was to identify the potential culprit group responsible for the internet outage by comparing a template graph with five candidate graphs provided in the problem statement. The challenge involved using data visualization to determine the candidate graph that closely resembles the template graph.

This group project employed technologies such as D3.js [2] for data visualization, HTML, CSS, and Bootstrap for the front-end dashboard, along with GIT for version control to create a web-based visualization application. D3 visualizations were instrumental in assessing graph data similarity through the comparison of five visualization types: pie chart, donut chart, stream graph, heat map, bar chart, network diagram, and world map chart. Additionally, Interconnecting all five visualizations and developing an innovative visualization view was a fundamental requirement of the project, accomplished through two approaches:

1) Utilized a pie chart for streamlined data filtering and enhanced comparison.
2) Enhanced the network diagram by incorporating a hover-triggered donut chart for detailed information display.

II. Explanation of Solution

The solution formulation involved a systematic approach, encompassing several crucial components. The initial phase consisted of utilizing Five Design Sheets [3] to create preliminary sketches of visualizations and discerning optimal solutions for the given problem. This process required extensive brainstorming and idea filtration.

Subsequently, project deadlines were established, and data-cleaning procedures were implemented to ensure the preparation of a refined dataset suitable for visualization. The foundational structure for interlinking and data filters was then defined.

Moving forward, the coding phase encompassed the development of individual visualizations, incorporating key elements such as interactions, animations, legends, marks, and channels. This stage aimed at not only presenting the data effectively but also enhancing the user experience through thoughtful design and functionality.

The overall methodology followed a rigorous sequence of steps, emphasizing careful consideration and strategic planning at each stage of the project.

A. Dataset Description

The dataset includes node-to-node relationships, where nodes represent individuals or items, and the relationships indicate communication or procurement channels. The utilization of start time and end time parameters provided in the dataset revealed that the data spans a one-year duration. This deduction was instrumental in defining the domain and range for the visualizations. During the data cleaning phase, the relationships and node data were condensed for a more streamlined visualization. The six types of relationships were condensed to four, ensuring no loss of data. The resulting dataset exclusively includes only the person nodes, and their relationships are aggregated and depicted with a single edge between them. In addition to relationship data, the dataset incorporates demographic information for each node, reflecting the spending and income habits of individuals. Initially consisting of around 30 categories, this demographic data was streamlined to 11 for the bar chart and further reduced to just 4 for the donut chart visualization. Spatial and temporal data for each node are also provided for comprehensive analysis.

B. System Description

Initially, when no filter is selected, the user is presented with aggregated data, with the chosen aggregation highlighted in blue as shown in Fig 1. For a side-by-side comparison, the template and network diagram are positioned adjacent to each other. Filters are located at the top using the pie chart, stream graph, and heat map, where the stream graph and heat map incorporate time filters, while the pie chart includes communication channel filters. To switch between the stream graph and heat map a toggle is provided. Additionally, the map chart and bar chart are displayed below the network diagram, showcasing spatial meetup points and the demographic signature of each person, respectively.

C. Network Diagram

A network diagram is employed to visualize both the template and candidate networks, as depicted in Fig. 2. Node size in the diagram corresponds to the number of edges coming in and going out, while edge thickness signifies the weight of the total edges between two nodes. Edge opacity is determined

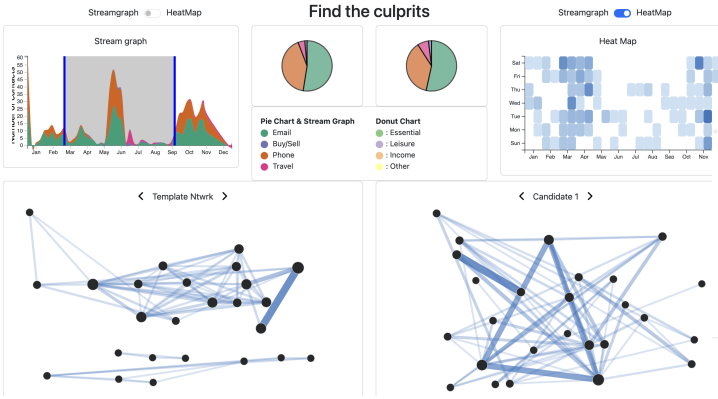


Fig. 1. Main view for the project.

by edge weights, with darker shading highlighting more crucial edges. Node positioning is based on node IDs and the D3 force layout. Additionally, users have the capability to switch between the networks, facilitating a seamless comparison.

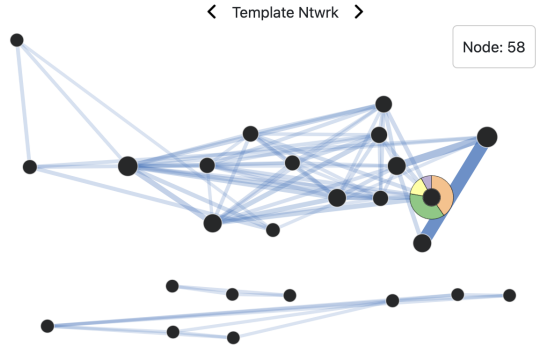


Fig. 2. Template network and donut chart using network diagram.

D. Donut Chart

Upon hovering over a node in the network diagram, a donut chart is displayed around the node, illustrating the aggregated demographic distribution of that individual as shown in Fig 2. The four categories represented in the donut chart are Essential, Leisure, Income, and Other. Leveraging these demographic signatures allows for the identification and comparison of the same nodes in both the template and candidate graphs. Additionally, a tooltip is presented, providing node ID information. This feature represents an innovative visualization aspect of the project.

E. Bar chart

The bar chart visualizes eleven demographic categories (Telephone, Education, Transportation, Income, Substances, Donations, Leisure, Healthcare, Personal, Living, and Household), initially displaying average spending and income habits. However, upon clicking a node in the network diagram, the bar chart dynamically updates to illustrate the comparison specifically between the clicked nodes as shown in Fig 3. This chart facilitates a comprehensive demographic analysis, enabling a detailed comparison of node-level demographic data.

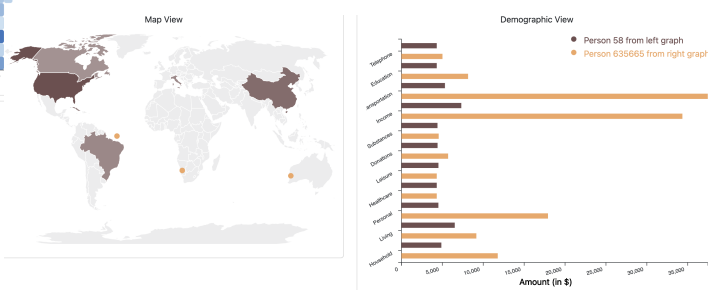


Fig. 3. World Map chart and Bar chart.

F. World Map Chart

The map chart highlights common meetup points across five countries, with opacity indicating the total number of meetups and points representing individuals' spatial data as shown in Fig. 3. This information dynamically updates when changes are made to the network diagram. A tooltip presents additional information about the country or point upon hover.

G. Pie chart

The pie chart in Fig. 4 illustrates the distribution of four communication channels: email, phone, travel, and buy/sell. Hovering over each sector displays the total count for that channel. Additionally, the chart serves as a channel filter. Clicking on a sector filters the data, updating all visualizations to exclusively show the selected channel type, accompanied by a change in color representation.

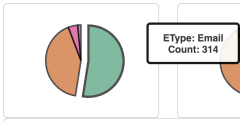


Fig. 4. Pie chart representing communication channels.

H. Stream graph

The stream graph illustrates the weekly temporal distribution of each communication channel, also serving as a time filter when the window size is adjusted as shown in Fig 5. Clicking on a specific stream reveals the data for that stream and the selected time window, and all visualizations are updated accordingly.

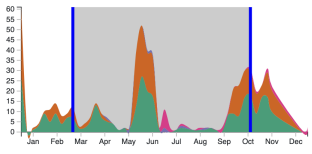


Fig. 5. Stream graph representing week temporal data.

I. Heat Map

The heat map offers enhanced temporal granularity by presenting a day-wise distribution of communication contact on each day as shown in Fig 6. When hovered over a cell the tooltip displays the frequency of communication.

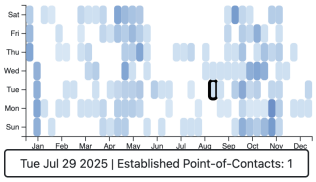


Fig. 6. Heat map representing daily temporal data .

III. Results

A thorough analysis was performed, comparing pie chart ratios, stream graph peaks, and heat map signatures for both template and candidate graphs. While the stream graphs for candidate graphs exhibit a slight offset to the left, overall, they appear similar. Similarly, the heat maps show a slight offset in values, indicating elevated communication on specific days. Hence, the results indicated that Candidate 1 and Candidate 3 share the highest similarity with the template graph mentioned in the problem statement.

To support these findings, the network diagram was scrutinized, taking into account the comparison of edge strengths, the demographic signature distribution of nodes on the donut chart, and lifestyle habits as indicated by the lengths of the bar chart. These additional examinations collectively strengthened the conclusion that individual nodes in the graphs of Candidate 1 and Candidate 3 closely correspond to the provided template graph in the problem statement.

Furthermore, the spatial data was identified as random noise and deemed unreliable for visual analysis after plotting.

IV. Individual Contributions & Learnings

I contributed to project planning, research, implementation, and fine-tuning, applying class-taught principles to create five design sheets. After two iterations, I finalized the project's overall look and discussed it with my teammates. Recognizing the importance of project planning and workload distribution, especially for a sizable project with six visualizations, I took the initiative to distribute tasks in advance.

I played a key role in developing the network diagram [4], implementing the force layout, and ensuring its accurate rendering with dynamically changing data. The learning curve involved fine-tuning the force between nodes, adjusting filtered data to display only one edge between nodes, and managing the force accordingly.

Additionally, I contributed to the World Map Chart [5] by randomly selecting countries and aligning data with specified IDs. Important learning here was offsetting points representing nodes to overlay them accurately on continents rather than oceans.

My involvement extended to front-end elements, including the write-up, legends, explanations of marks and channels, arrow buttons for switching network diagrams, and the toggle for stream graph and heat map. The learning in this area emphasized concise explanations, visually appealing content arrangement, and ensuring proper functionality of buttons and toggles. I also integrated tooltips into the network diagram, map chart, and bar chart for easy identification and enhanced user experience.

I learned D3.js and JavaScript in-depth. I also applied a cohesive color scheme to streamline visualizations, ensuring consistency in the project's development. Managing intricate design principles, such as tooltips, legends, color contrasts, element positioning, animations, and intuitive interactions, presented challenges. With the help of five design sheets, I tracked these intricacies across all visualizations throughout the project.

Additionally, I acquired a profound understanding of GIT, encompassing its commands and branch management. Reviewing my teammates' code and pull requests on GitHub taught me how to proofread someone else's code and ensure the main branch remains functional after merging a pull request. This experience instilled in me the importance of applying industrial practices for code development and maintenance, emphasizing modular coding with functions. I also learned to code by creating branches and ensured that my branch was always up to date with the main. This approach significantly reduced merge conflicts during the merging process.

Overall, my involvement encompassed diverse aspects of the project, from project planning to network diagram implementation to front-end elements development, demonstrating a comprehensive skill set in both technical and collaborative aspects of the project.

V. Team Members

Name	Email
Vishwesh Pillai	vpillai9@asu.edu
Parth Shah	prshah11@asu.edu
Yadvendra Naveen	ynaveen@asu.edu
Devdutt Oruganti	dorugant@asu.edu
Avi Mehta	amehta65@asu.edu

References

[1] "Mini-Challenge 1: Graph Analysis," https://vast-challenge.github.io/2020/MC1.html.

[2] "D3 by Observable | The JavaScript library for bespoke data visualization," https://d3js.org/.

[3] J. C. Roberts, C. Headleand, and P. D. Ritsos, "Sketching Designs Using the Five Design-Sheet Methodology," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 419–428, jan 31 2016.

[4] M. Bostock, "Force-directed graph, disjoint," Jun 2023. [Online]. Available: https://observablehq.com/@d3/disjoint-force-directed-graph

[5] Y. Holtz, "Basic background map in d3.js." [Online]. Available: https://d3-graph-gallery.com/graph/backgroundmap$_{b}asic.html$