It's a Jungle Out There: Data Abstraction Elephants and Mental Models

IT'S A JUNGLE OUT THERE: DATA ABSTRACTION ELEPHANTS AND MENTAL MODELS

by

Kathryn Williams

A Dissertation Submitted to the Faculty of the

DEPARTMENT OF COMPUTER SCIENCE

In Partial Fulfillment of the Requirements
For the Degree of

DOCTOR OF PHILOSOPHY

In the Graduate College

THE UNIVERSITY OF ARIZONA

2023

THE UNIVERSITY OF ARIZONA
GRADUATE COLLEGE

As members of the Dissertation Committee, we certify that we have read the dissertation prepared by: *Kathryn Williams*, titled: *It's a Jungle Out There: Data Abstraction Elephants and Mental Models.*

and recommend that it be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy.

[illegible]
Katherine E Isaacs (Apr 17, 2023 17:22 MDT)
Dr. Kate Isaacs Date: Apr 17, 2023

Joshua A. Levine
Dr. Josh Levine Date: Apr 11, 2023

[illegible]
Dr. Carlos Scheidegger Date: Apr 11, 2023

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertation to the Graduate College.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it be accepted as fulfilling the dissertation requirement.

[illegible]
Katherine E Isaacs (Apr 17, 2023 17:22 MDT)
Dr. Kate Isaacs Date: Apr 17, 2023
Dissertation Committee Chair
Computer Science Department

ACKNOWLEDGEMENTS

I would like to thank my family and friends for believing that I'm smart and can do this, even when I do not feel like I'm smart or that I can do this.

A huge thank you to my advisor Katherine (Kate) Isaacs and my committee members Joshua (Josh) Levine and Carlos Scheidegger. You all supported my ideas, provided advice, and were always willing to take extra time to explain and debug. Kate, thank you for allowing me to take a year-long break during my Ph. D and supporting my version of work-life-swim balance.

Thank you the members of the Phylanx Team, APEX Group, and Ste||ar groups for supporting and using Atria, and continuing to collaborate with Kate, Connor, and Sayef.

Thank you to the conference reviewers for their thoughtful suggestions on each submission. Thank you to the participants in my studies for their creativity and willingness to draw and explore alternative ways of thinking.

DEDICATION

This dissertation is dedicated to Dr. Laurie Heyer, for believing in my research career at the very beginning.

LAND ACKNOWLEDGEMENT

We respectfully acknowledge the University of Arizona is on the land and territories of Indigenous peoples. Today, Arizona is home to 22 federally recognized tribes, with Tucson being home to the O'odham and the Yaqui. Committed to diversity and inclusion, the University strives to build sustainable relationships with sovereign Native Nations and Indigenous communities through education offerings, partnerships, and community service.

TABLE OF CONTENTS

TABLE OF CONTENTS – *Continued*

TABLE OF CONTENTS – *Continued*

LIST OF FIGURES

LIST OF FIGURES – *Continued*

LIST OF FIGURES – *Continued*

LIST OF FIGURES – *Continued*

LIST OF TABLES

ABSTRACT

This dissertation explores how mental models of data influence visualization design. A mental model of a dataset is the user's understanding of the data, encompassing their prior experiences, interests, and knowledge of the data. In this dissertation, I demonstrate how the flexibility of mental models can allow for changes in the visualization, while paradoxically the inflexibility of personal mental models indicates designers should prioritize aligning the visualization with the mental model. To connect the internal (i.e., mental models) with the external (i.e., data visualizations), we suggest using data abstractions to link the mental model to a data-related structure. This gives the user and designer a common language to start from and guides visualization design.

Choosing the data abstraction has been recognized as an important part of the design process, but this abstraction is typically based on the data itself, not on users' mental models of the data. Mental model research has been centered around mental models that arise after seeing a visualization and how users utilize their mental models of the existing visualization. This dissertation addresses this gap in the visualization design methodology from three perspectives: (1) how mental models of data are created from non-tabular data without prior visualizations, (2) describing data abstractions and exploring alternatives, and (3) how changing data abstractions requires changing the visualization to fit the mental model.

I first describe a design study of a tree visualization in which my collaborators and I considered changing the data abstraction to fit the data, but ultimately chose the abstraction that fit our users' mental models. Second, I explore types of data abstractions, how they are related, and the effects of changing data abstractions. Finally, I investigate how different mental models can arise from the same dataset via a study involving sketching non-tabular data. I conclude by asking how to better facilitate these discussions of mental models before and during a visualization

design study. Understanding and formalizing what questions to ask to elicit useful descriptions of mental models will allow designers to create visualizations that better suit users' mental models of the data.

CHAPTER 1

Introduction

Visualization designers typically focus on the design of external representations of data, e.g., charts and graphs that users take in and interpret to gain an understanding of a dataset. However, there is an important piece that visualization designers often leave unexplored: their users' internal data representation, i.e., their *mental model* of the dataset [69, 62]. A user who regularly interacts with the same or similar data has a mental model of the data, encompassing their background knowledge on the subject, intuition, and knowledge about the underlying structure of the data and possible connections. Even a user approaching a novel dataset will have experiences, knowledge, and expectations that shape how they look at the dataset. Since the user is an expert on navigating the dataset, they can provide examples for the visualization designer, warn when data values are illogical, and explain how they make decisions using the data. All of these facets of information are useful for the visualization designer, helping them avoid data errors and understand the tasks the user prioritizes. By simply basing the visualization design process on the dataset, visualization designers are missing out on the wealth of expertise in their users' mental models.

Mental models of visualizations have gained significance and discussion in recent years. Liu and Stasko reviewed cognitive science and HCI literature to develop a definition of mental models in the context of information visualization [91]. They define a mental model as "a functional analogue representation to an external interactive visualization system." Prior work describes how users construct mental models of external representations, like multi-view visualizations [131] or a photo-sharing website [137], and suggests ways to measure mental models of information visualizations [92]. Work by Lewis in 1986 even provides guidelines for how to effectively construct mental models of tasks performed on computers [88]. Extensive work by Götschi and Sanders et al. evaluates students' mental models of recursion [52, 128, 130], assessing how well teaching methods help students build accurate mental models of recursion. These mental models are all internal representations of external visualization systems, or rely on external visualizations and instruction to develop a mental model of a concept. In this dissertation, we begin by discussing mental models of concrete visualizations, then move toward more abstract representations of data until we end with mental models that arise from data in paragraph form, without any existing visualization.

Even when the datasets are identical, two people viewing the data will not create the same mental models. Each individual has a unique background on the subject of the dataset, prioritizes different attributes, and connects with various aspects of the data. Both users have unique perspectives about the data — what data points and tasks are interesting, what connections between points are significant, and how to meaningfully group the data. These differences create tension between two mental models, which is not necessarily bad or good. The commonalities between the mental models may connect two users and the differences may be areas of discussion and exploration.

Tension also exists between the imagined mental model and how the mental model can be physically represented in relational databases, spreadsheets, or other types of data storage. A *data abstraction* is a mapping of domain-specific data to an abstract data type [104], e.g., course data can be mapped to a table or to a network. It may be beneficial to identify the data abstraction that is most similar to the user's mental model, and then explore known transformations of the data abstraction to find a middle ground between what most aligns with the mental model, yet is also a practical way to store and interact with the dataset.

Choosing the data abstraction has been recognized as an important part of both the data design step [102] and the visualization design process [103]. Still, this abstraction is typically based on the data itself, not on users' mental models of the data. Bigelow et al. emphasize that designers need to be able to define and modify data abstractions to match their users' mental models [10]. A poor choice of data abstraction can delay progress and the final visualization may not end up supporting the user's needs [103].

This dissertation explores how mental models and data abstractions influence visualization design. Specifically, my collaborators and I demonstrate how the flexibility of data abstractions can allow for changes in the visualization, while paradoxically the inflexibility of personal mental models indicates we should prioritize the abstraction that most closely matches the mental model. Our overarching research goal is to find ways to align the data abstraction and visualization design with the user's mental model. Problems arise when we do not, causing conflicts when discussing how to collect, store, and visualize the data.

This dissertation addresses this gap from three perspectives: (1) how we maintain the mental model in a visualization among changing data and abstractions; (2) describing existing data abstractions and exploring alternative, uncovered yet useful data abstractions (what we call *latent data abstractions*); (3) what kinds of data abstractions appear in mental models when these mental models arise from non-tabular data.

In chapter 3, I describe our design study that resulted in Atria, an interactive tree visualization. Atria is used for performance analysis of task-parallel programs, a specific type of runtime used for distributed computing on high-performance systems. We discovered well into the construction of Atria that we had misjudged our underlying data abstraction — what we thought was a tree was instead a graph. We describe how we adapted the design study methodology [133] to the "moving target" of both the data and the domain experts' concerns and how this movement kept both the visualization and programming project healthy. We explain how our overarching team goal helped us and our collaborators remain motivated and navigate our incorrect abstraction choice.

Next, in chapter 4, we study types of data abstractions, how they are related, and the effects of changing data abstractions. We set out to uncover how malleable data abstractions are and to better understand the process of pursuing latent data abstractions. A data abstraction is considered latent when the data abstraction is meaningful and useful, yet undiscovered. It has yet to be fully elucidated, communicated, documented, and formatted. We surveyed and interviewed a wide assortment of data workers about how they classify their data, selecting one of six abstractions. We then asked them to consider their data in an alternative data abstraction, exploring how the qualities of the data would change and how their analysis would change. We used grounded theory methodology to generate codes and themes from the responses. Our choice of data abstraction typology framed the topic for our participants, eliciting rich communication and reflection about data. We provide guidelines for developing data abstractions and for probing latent data abstractions.

Finally, in chapter 5, we investigate "data elephants", an analogy we use to illustrate how people can create different mental models from the same non-tabular dataset. In semi-structured interviews, participants were given one of three datasets and were asked to draw their mental model of the data. We discussed with participants their sketch and mental model, the source of their idea, and whether their mental model of the dataset evolved during drawing. We used the principle of surprise and saturation, critical components of grounded theory, to guide the questions asked in our interviews and the direction of our study. Our participants used diverse language and abstractions to represent their mental model; this diversity was influenced by several factors including recent examples the participant had seen, imagined purpose-seeking tasks, and their definition of what "data" is.

1.1 Contributions

This dissertation makes the following contributions:

- A design study of how to proceed in the face of a shifting data abstraction and changing data needs.
- Guidelines for pursuing and revealing latent data abstractions when visualization designers collaborate with data workers.
- Implications for visualization designers on eliciting and understanding their user's mental model.

At the end of each chapter, I include a"Family and Friends Summary" as a way to make this dissertation accessible to those who love and support me the most. For academic readers, this section is optional. It summarizes the chapter in a more casual way.

Family and Friends Summary: When you read or view a dataset, you have your own background and intuitions about the data and subject matter. We call this unique viewpoint your *mental model* of a dataset. Trouble and confusion arise when a visualization of the dataset uses a data abstraction (i.e., an underlying structure of the dataset) that does not match your mental model. In this dissertation, I show how my collaborators and I had to change data abstractions but kept the overall visualization in alignment with our users' mental models. I then discuss how malleable these data abstractions are: can we change from one data abstraction to the other? How helpful is this change? Finally, I connect these data abstractions to sketches of mental models and how our participants think about data.

1.2 Reproducibility Table

To embrace completeness and transparency, I make my collected data, code, and analysis process available to other researchers to support reproducibility of my methods. Table 1.1 contains links to the code repositories and data repositories used in the making of this dissertation.

Project (Chapter)	Resource	Location
Visualizing a Moving Target: A Design Study on Task Parallel Programs in the Presence of Evolving Data and Concerns (chapter 3)	Atria tree visualization code	`https://github.com/hdc-arizona/traveler-tree`
	Traveler dashboard: subsequent work incorporating Atria tree visualization by Sakin et al.	`https://github.com/hdc-arizona/traveler-integrated`
Guidelines for Pursuing and Revealing Data Abstractions (chapter 4)	Interactive collection of survey responses	`https://alex-r-bigelow.github.io/wrangling-survey/Responses.html`
	Repository of codes, themes, and supplemental material	`https://osf.io/382fn/`
Data Abstraction Elephants: The Initial Diversity of Data Idioms and Mental Models chapter 5	Collection of interview transcripts and sketches	`https://osf.io/kvnb9/`
	Repository of codes	`https://github.com/kawilliams/mental-models-codes`
	Jamboard	`https://kawilliams.github.io/papers/elephants_supplemental/Jamboard.pdf`

Table 1.1: A list of the raw data and resources used in this dissertation, including their url locations. Additional information and supplemental material for each paper can be found at my personal website: `https://kawilliams.github.io/`

CHAPTER 2

Related Works

For this dissertation, I returned to several works on design studies, task abstractions, and data abstractions. These works are described in detail in this section and are referenced in chapters 3, 4, and 5. Relevant related work that is specific to each chapter is located in the Related Work section of that chapter (e.g., chapter 3 contains Related Work on execution graph visualizations).

2.1 Design Study and Task Abstraction

Throughout this dissertation, discussions of data abstractions and mental models are primarily to benefit visualization experts in the early stages of the design study framework [133]. Sedlmair et al. define a *design study* as "a project in which visualization researchers analyze a specific real-world problem faced by domain experts, design a visualization system that supports solving this problem, validate the design, and reflect about lessons learned in order to refine visualization design guidelines." Key in this process is the gain for both the domain experts and the visualization researchers: the visualization becomes a deliverable for the domain experts and the process becomes guidance for the visualization community. The design study methodology consists of a nine-stage framework. These stages are *learn, winnow, cast, discover, design, implement, deploy, reflect, write* (for a complete description of each stage, see the original work [133]). The authors describe *pitfalls* that may occur within each stage of the framework. The framework is not strictly a linear progression through steps; instead, it incorporates and encourages full cycles and sub-cycles in the design study process. I primarily use this framework to situate our work in 3 and chapter 5.

During the *design* stage in the design study framework, the visualization designer generates data abstractions, visual encodings, and interaction mechanisms to use in the visualization. I elaborate on *data abstractions* below in section 2.2. The choices made by the visualization designer are based on the tasks that the user intends to accomplish through using the visualization. To bridge low-level visualization tasks to more complex goals, researchers have proposed compositions of tasks and hierarchical task analysis with visualization *task abstractions* [84, 3, 4, 18, 87, 123, 126]. Task abstractions help visualization designers understand the tasks their users want to prioritize and reinforce how the design of the visualization can accomplish these tasks. In the design study of Atria, my collaborators and I linked our tasks to goals in multiple levels, similar to Zhang et al. [162] (see Figure 3.2).

Throughout the design study of Atria (chapter 3), my co-authors and I relied on flexibility and collaboration for our success, even though we embarked on the project in the face of known pitfalls [133]. We built Atria as a simple and adaptable technology probe [63]; this drove discussions about what data needed to be collected and aspects of the data that had not been clarified. We deployed our probe in diverse technical and practical contexts with the runtime and performance teams, creating a parallel, multi-channel collaboration environment like that described in detail by Wood et al. [157].

2.2 Data Abstractions

Prior to any visualization, data workers actively work with data and this process impacts how the data arrive to the visualization expert. Muller et al. [102] describe five stages that data workers apply when working with data: discovery, capture, curation, design, and creation. This work focuses on designing data, specifically the data abstraction [98].

A *data abstraction* is a mapping of domain-specific data to an abstract data type [104], e.g., power station supply lines can be mapped to a network. A *latent data abstraction* is an alternative abstraction of the same data that has yet to be elucidated and may provide a different, useful perspective, e.g., power station supply lines as a table displaying amounts of power produced. Feinberg observes that the mere use of a dataset makes the user a designer of its abstraction [42], even if users are unaware of the inherent flexibility in the dataset. The designers used this flexibility in graph abstractions and creativity when encoding genome sequences for the ABySS-Explorer [108]. Likewise, I describe how we demonstrated this malleability through a study in chapter 4. Communicating about data abstractions is difficult [119] since the user may not understand how to describe the tasks they want to perform or the goals they are trying to accomplish. Discussions about options for the dataset and abstraction should happen early in the design study process and should be frequently re-examined by the domain experts to ensure correctness and cohesion with their mental model of the problem [133].

The visualization designer may need to modify the abstraction based on their understanding of how the user interacts with the data and the tasks they are trying to accomplish. Often there is not a single correct abstraction; instead abstractions must be designed [98, 102] to best suit the user's needs. When designing data, too much focus on a single data abstraction has been observed to limit creativity [10]. Consequently, there is a need to learn to develop a "data vision" to exercise discretion and creativity in designing abstractions [112].

Creativity workshops can help avoid a single-minded pursuit of a data abstract and encourage multiple perspectives [51, 77]. Prior work in creative visualization workshops and collaborative prototyping provide a framework for facilitating exploring alternative, useful design ideas [77, 39, 40]. As such, sketching is a low-cost, popular method and one that we use in our study in chapter 5; I provide an overview of prior uses of sketching in visualization below (section 2.3).

2.3 Mental Models and Sketching

A person's *mental model* is their a personal understanding of a topic that may consist of representations of objects, background knowledge about the topic, and connections to related topics. In our context, a mental model of a dataset includes the person's knowledge of the contents, prior experience working with the dataset, and their intuition about the data and relationships within the dataset. Originating in cognitive science [69], mental models have been explored in human-computer interaction and visualization [14, 55, 92, 131, 137]. Liu and Stasko reviewed the literature and defined mental models in the context of information visualization [91]. Research on mental models of multi-view visualizations [131], social networks [137], and even the concept of recursion [52, 128] have all been examined. While a diverse range of topics, these mental models all start as external images, from visualizations to lectures with slides to diagrams. Mental models can evolve, like when learning the new topic of recursion, or through interaction with the subject, like when users interact with and understand a visualization system [14]. These prior works focus solely on the mental model that is created *after* viewing an existing visualization– we are concerned with mental models created from the dataset before viewing any influencing abstractions or visualizations.

Perceiving external representations, such as visualizations or self-generated sketches, allows people to think more powerfully than without those representations [152]. Bartram et al. found that data analysts' annotations within spreadsheets allowed users to gain a closer understanding of the data through the analytics processes and augmented sense-making [5]. Externalizing our thoughts allows people to work through operations that are too complicated to do internally, handle more complicated structures, and run processes more quickly and with more precision than when executed strictly internally [79].

Sketching is a simple way to externalize inner thought processes and mental models, such as students' concepts of time [44] or homeowners' concepts of their home wireless network [118]. Understanding the language of diagrams and how we visualize our thoughts [143] enables us to successfully collaborate and share visualizations. In collaborative environments, sketches can be created spontaneously on whiteboards or paper during brainstorming, thinking, communicating, and general problem-solving [148, 124]. Alternatives to sketching and to using Microsoft Excel, such as tangible tokens [61, 60, 158] or personal physicalizations using craft supplies [138], have also been proposed as ways to quickly visualize datasets with minimal instruction.

Family and Friends Summary: I define components of the visualization design process, specifically the design study methodology. A *design study* is a data visualization project that not only produces a useful visualization tool for the users, but also generates insights and knowledge for the visualization designer to share with the visualization research community. The *design study methodology* is a framework for conducting a design study, which includes guidance before the visualization design, during the data and visualization creation, and in the post-project analysis phase. Once the visualization designer has the data, they select a data abstraction to guide the construction of the visualization. A *data abstraction* is a generalized framework of data (e.g., a table) rather than a specific dataset (e.g., a table of foods and their caloric and nutritional breakdowns). A visualization designer knows of several tools and techniques for each kind of data abstraction so the ability to generalize from a specific dataset to a general data abstraction makes for less work for the designer.

In this dissertation, we consider trying to match the data abstraction with the user's mental model, rather than solely with the data. A person's *mental model* of a dataset is the combination of their knowledge about the data, their prior experience working with the dataset, and their intuition about the dataset. With the table of food example, even though the data might be presented in a tabular data abstraction, the user might prefer thinking of the food data in groups, like breakfast foods, vegetables, or vegetarian options. Thus, a grouped data abstraction may best fit the user's mental model, which calls for different visualization design choices.

CHAPTER 3

Visualizing a Moving Target: A Design Study on Task Parallel Programs in the Presence of Evolving Data and Concerns

I begin this discussion of mental models and data abstractions with a concrete example. In this design study, the user's mental model helped ground an ever-changing visualization design. In this chapter, I discuss how my collaborators and I designed a tree visualization throughout shifting data concerns. These shifts meant we had to continuously adapt our tree visualization design and implement feedback from our users. I started prototyping data visualizations using a small tree dataset example of the execution of a computer program. However, construction of our tree visualization was momentarily halted when our collaborators notified us that what we thought were trees were actually graphs. In computer programs, functions call smaller, sub-functions and these caller-callee relationships typically form a tree. However, recursive functions can occur, meaning an edge in the tree returns to its root node, turning the tree into a graph. From a visualization design perspective, this change in data abstraction from a tree to a graph would have implications for the layout of the graph layout and would break the existing visualization program.

During brainstorm sessions of different design options, we learned that our users primarily think of the data as a tree and implicitly understand where graph edges may occur. They had been using the tree visualization in their workflow and appreciated the cleanliness of the tree layout for the computer program data. Our users had a sufficient understanding of the programs they were using so they could mentally "fill in" the missing edges in the graph, without them actually existing. These graph edges are still important, however, so we proposed changes to the visualization that allowed our users to see this underlying graph structure while also preserving much of the existing tree visualization that both the users and visualization researchers had grown to rely on.

This work was previously presented at IEEE VIS 2019 and the published work is available [156][1]. The content of the original paper has been modified slightly to fit the flow of this dissertation. This work was done by Alex Bigelow, Katherine (Kate) Isaacs, and myself. Our collaborators were the members of the Phylanx group[2], consisting of Rod Tohid, Bibek Wagle, Shahrzad Shirzad, Patrick Diehl, Adrian Serio, Alireza Kheirkhahan, Parsa Amini, Hartmut Kaiser, and Kevin Huck. My Atria tree visualizations have continued to be used in more elaborate tools, including Jupyter notebooks [82] and an interactive linked dashboard [19, 125]. Of note, the dashboard and work by Sakin et al. [125] explicitly extends the Atria task analysis to temporal data, expanding the impact of my original work and allows our users to analyze the program performance events with a Gantt chart timeline.

This work was supported by the United States Department of Defense through DTIC Contract FA8075-14-D-0002-0007 and by the National Science Foundation under NSF III-1656958.

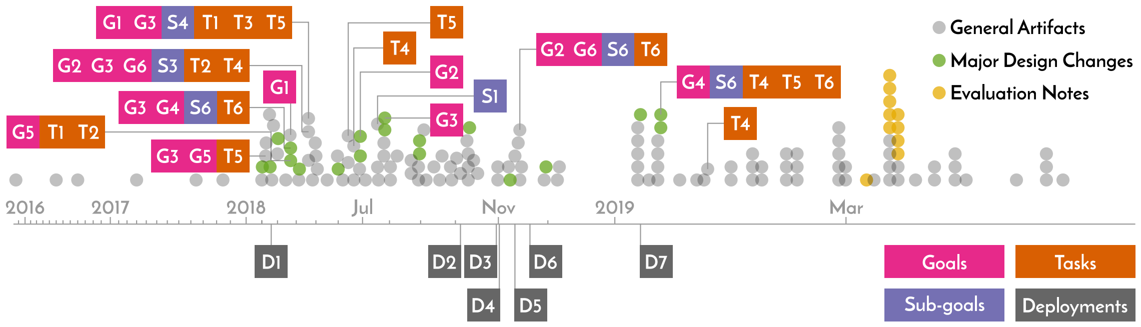


Figure 3.1: Design Study timeline (log scale). The top contains a mark for each collected artifact. Connections to identified goals, sub-goals, and tasks are marked when direct evidence for them has been identified. Artifacts from meetings presenting major design changes and notes from the evaluation sessions of Section 3.6.2 are indicated with color. The bottom shows the timing of various deployments with users. This rich collection of over 150 artifacts mitigated issues in designing around shifting data and concerns.

[1]`doi:10.1109/TVCG.2019.2934285`

[2]Phylanx repository: `https://github.com/STEllAR-GROUP/phylanx`

3.1 Introduction

When choosing whether to move forward with a design study, there are several questions a visualization expert should answer to ensure project viability [133]. Among those questions are: (1) whether real, non-synthetic, data is available, and (2) whether the tasks that domain experts will use the visualization for will persist long enough to complete the study. Ensuring these points can help avoid problems arising in designing for the wrong data assumptions or having the users lose interest before the system is completed and evaluated. While in most cases it is prudent to avoid these problems, we argue there are circumstances in which it is fruitful to accept them.

In particular, there are scenarios where precisely what data to collect is an open question. The answer would ideally be driven by what analysis needs to be performed. This "chicken and egg" situation can dissuade both domain and visualization experts from engaging. The domain expert does not want to collect data with no plan for analysis. The visualization expert cannot act without real data. Thus, an opportunity for visualization to inform the data collection process is lost and the greater problem remains unsolved.

We observed this scenario in the domain of task-parallel computing. Parallel systems are challenging to comprehend due to their complexity. Several factors affect performance and correctness including program source code; the parallel libraries used; the input to the program; and the architecture of the cluster on which it is run. Understanding the emergent behavior in these systems is necessary to optimize and debug them. While some areas of parallel computing have a long history of data collection for performance analysis, the data necessary to analyze a more recently prominent model—asynchronous many tasks—is a relatively open area.

Recognizing the pitfalls of changing data and concerns, but also the potential of using visualization to help drive the development of those data and concerns, we proceeded with the design study. Although the pitfalls could have derailed the project, we found other factors—such as the shared interest in the data collection problem and the identification of key recurring abstract structures—resulted in the creation of visualizations that were beneficial even as things changed. From these experiences, we demonstrated how the benefits of visualization are not only the design of the final solution, but the integration of the visualization team and its effects on the overall project development.

We describe our iterative design process and how we adapted the design study methodology [133] to the "moving targets" of our data and user tasks. Our task analysis and abstraction were developed through multiple rounds to account for evolving concerns.

Through this process, we developed our technology probe [63], Atria, a multi-view system for exploring execution graphs. Unlike other execution graph visualizations, we display the graph as an expression tree, evoking the main logical dependencies of the computation while preserving additional edges on demand. Atria provides the context of source code to the execution graph not only through linked highlighting, but also through a default aggregation and de-cluttering scheme based on line of code.

We further discuss how the changing concerns affected deployment with implications for design, particularly in the case of Jupyter notebooks [82]. We augment our assessment of our design with evaluation sessions and semi-structured interviews.

The major contributions of our study are as follows:

- A task analysis for execution graphs (Section 3.4) and how we iterated on that analysis under changing conditions,
- The design of Atria, a interactive visual tool for analyzing task execution graphs (Section 3.5),
- A discussion of the design adaptations and the difficulties of incorporating visualization within the Jupyter notebook environment (Section 3.5.4), and
- Reflection on the project and recommendations for conducting design studies in the presence of evolving data and concerns (Sections 3.3.2, 3.3.2, and 3.7).

We discuss related work (Section 3.2) followed by necessary background in task-parallel execution graphs (Section 3.2.4). We then discuss the organization of the project (Section 3.3.2). We conclude in Section 3.8.

3.2 Related Work

We discuss related work in design study methodology, task abstraction, visualization of execution graphs, and tree visualization techniques.

3.2.1 Design Studies

We report on the initial phases of an ongoing design study, including multiple deployments of our system, Atria, as a technology probe [63]. This process has enabled us to collect rich, qualitative data. While informing future iterations in our collaboration with domain experts, reflections [96] (Section 3.7) on these data already have meaningful implications for the visualization community.

In the context of the nine-stage framework for design study methodology [133], this work represents a full cycle, including many sub-cycles, of each of the nine stages. We build upon previous visualization and design study experience at the *learn* stage to inform careful, deliberate decisions at the *winnow* stage, that we discuss in Section 3.3.2. We observed refinements at the *cast* stage over time, where deployments and conversations with domain experts exposed deeper insights into the various roles that they play in practice. At the *discover* stage, our tool enabled insights for our collaborators and ourselves, particularly with respect to horizontal movements in the task-information space—Atria drove many discussions about what data needed to be collected that were unlikely to have occurred without our probe's involvement. The simple nature of the tool enabled relatively simple *design* refinements, as well as rapid *implement* and *deploy* stages. Throughout we use the term *iteratively* to describe the repetition of design stages and *iteration* to mean each new version of Atria presented to our collaborators, as marked in green in Figure 3.1.

In framing our contributions in terms of the nine-stage framework, it is important to address considerations that it does not capture. We deployed Atria in very diverse technical and practical contexts, creating a parallel, multi-channel collaboration environment like that described in detail by Wood et al. [157]. We did not experience the constraints described by Crisan et al. [33], but our process also benefited from thorough artifact generation, frequent communication, and staged design. Our focus in this early phase of a long-term collaboration has been to elicit robust design requirements, rather than deploying prototypes too early [94]. Instead, we report on the use of a technology probe to build, intervene, and evaluate [93], with Sedlmair et al.'s [133] *reflect* and *write* stages applying continuously. As Hinrichs et al. [59] show in digital humanities, we demonstrate value in the visualization *process*. Finally, we contend that, in some cases, it may be beneficial to consider collaborations where a design study will have an opportunity to impact how data is collected and how the initial data abstraction is designed.

Task Abstraction. Several methods have been proposed for bridging low level visualization tasks to more complex goals [18, 123, 84, 162]. Brehmer and Munzner [18] propose composition of low level tasks. Zhang et al. [162] demonstrate combining hierarchical task analysis [3, 4, 126] with visualization task abstractions. As the data and concerns evolved, we did not observe stationary tasks at a low enough level to yet apply these detailed methods meaningfully. However, recognizing the importance of linking goals and tasks, we describe our task findings in terms of multiple levels. The lower level tasks of high level goals often overlapped, resulting in a lattice as described in Section 3.4.

3.2.2 Execution Graphs

Execution Graph Visualization. Node-link diagrams are prevalent in execution graph visualization [20, 38, 64, 121]. Dokulil and Katreniakova [38] remove edges that can be reached by other paths, plotting their results with `dot` [46, 47]. DAGViz [64] and Grain Graphs [121] use aggregation schemes to decrease the number of marks shown, taking advantage of nesting structures inherent to fork-join models of parallelism, but not present in general tasking models such as ours. Neither solution is interactive. In contrast, our interactive visualization abstracts execution graphs to a tree and aggregates based on the source code.

Trace data from tasking models has been visualized with Gantt charts. Ravel [65] shows the edges in Charm++ traces [74], but these are a subset of those in the full execution graph. Haugen et al. [56] show edges connected to a single task on demand. Pinto et al. [116] do not show edges. We do not have trace data and thus Gantt charts are inappropriate for our use.

Parallel calling context trees (CCTs) describe caller-callee relationships and are frequently visualized [2, 1, 89, 107, 7]. They differ from our execution graphs which are at a finer-grain task level and include dependencies not captured by CCTs. For a survey of visualizations across parallel computing models, see Isaacs et al. [67].

3.2.3 Trees Visualization

Visualizing a graph as a tree is an established practice [54, 41, 68, 110, 109] to reduce complexity and improve readability, especially when the tree has a semantic meaning. In our case, we visualize the edges relating to how the computation is expressed in code—its *expression tree*.

To reduce clutter, we collapse subtrees and encode them as triangles, like Space-Tree [117], but our collapsing strategy is based on meaning in the source code rather than screen space. Many techniques [85, 25, 117, 106, 141] exist for scalable hierarchy visualization. As part of the strategy of handling evolving data and tasks, we aim to "[satisfy] rather than optimize" [133] our depiction, but could apply these techniques once necessary. For a survey of tree techniques, see Schulz [132].

3.2.4 Task Parallel Programs and Execution Graphs

Parallel and distributed programs utilize vast computational resources to produce results that are often too time-consuming, if not infeasible, on a single processor. Achieving these time benefits often requires careful consideration of performance on the part of the programmers. Understanding observed performance is difficult because of the complexity of the emergent behavior stemming from the source code and the systems on which they run. The systems include not only the hardware, but the *runtime* which dictates the execution of the program and the environment in which it runs.

Performance is affected by several factors including adaptive scheduling policies that are difficult to predict. Intermediate structures may be created by the runtime that are not apparent to the programmers. Even known structures may be hard to reason about, given the dynamism in the system.

Asynchronous tasking runtimes (ATRs) are a class of parallel runtimes that have been gaining interest for their potential to increase resource utilization and overcome performance bottlenecks inherent to other paradigms. However, due to their more recent prominence, support for performance analysis of ATRs is still developing.

An *Asynchronous Tasking Runtime* supports an *Asynchronous Many-Task* (AMT) execution model. Typically, these models divide and encapsulate work (computation) into units known as *tasks*. The runtime then schedules the task for execution on one of its distributed resources. The flexibility to move tasks between resources allows ATRs to take advantage of parallelism other models may not.

3.2.5 Execution Graphs

Common to tasking models is the notion of an *execution graph*. In an execution graph, each task is a node. The edges are dependencies between tasks. A task cannot be executed until its dependencies are met. Tasks without dependencies between each other may run concurrently. Runtime developers are thus interested in these dependencies and their effect on scheduling decisions.

Execution graphs may be recorded during program execution. To reduce collection overhead, tasks of the same type (e.g., same function name) or with the same provenance (e.g., same function name and sequence of function names leading to the task) may be aggregated. We focus on the latter type of execution graph data in this project.

Attribute data is collected for each node in the execution graph. Typically for performance analysis, the number of times each task type was run (*count*) and the total time spent executing instances of that task are recorded. During our project, attribute data was augmented to also collect information about what mode the task was executed (see Section 3.3) and relation to line of source code.

3.2.6 Performance Data and Analysis in ATRs

There are many ATRs under active development [73, 74, 32, 6, 15, 36], but no standardized expectation for what performance-related data is collected. Existing parallel performance tools like TAU [134] and Score-P [83] can collect general performance data such as low-level profiles and traces, but often do not support ATR-specific data such as execution graphs. Existing execution graph work has often been specific to the ATR. Exactly what data could and should be collected when analyzing an ATR is an open area of research.

3.3 The Phylanx Project

We conducted this design study as part of a visualization initiative in the Phylanx Project. First, we provide technical background of the Phylanx [140] system necessary to understand our resulting visualization. We then discuss the organization of the project itself, how it led us to accept winnowing pitfalls, and the roles we cast in the design study.

3.3.1 Technical Overview of Phylanx

Phylanx is an actively-developed system for performing array computations in a distributed fashion. Its purpose is to provide the advantages of distributed resources (faster time-to-solution and the ability to scale beyond single-machine memory limitations) to data scientists while allowing them to use the tools with which they are familiar.

One such tool is Python. Phylanx has a front-end which allows data scientists to mark which of their Python functions they want run distributedly. Phylanx then translates the array operations in the Python code into HPX. HPX [73, 72] is a C++ standard library and asynchronous tasking runtime.

The average end users need not be concerned with how Phylanx transforms their code. However, power users interested in performance and the developers of the Phylanx system are.

Phylanx first translates the code into an intermediate representation in a domain-specific, functional language called PhySL. The function calls, control flow operations, data operations, and blocks found in the PhySL representation are referred to as *primitives*. These primitives are translated to tasks in the HPX runtime. The dependencies of each primitive are the arguments it needs to execute (which may be other primitives), data access operations, or any other constraints on variables the primitive uses. The dependencies and primitives form the execution graph which is run by HPX.

HPX can schedule any instance of a primitive in one of two modes: *synchronous* or *asynchronous*. A synchronous primitive is executed immediately from the primitive that initially spawned it. An asynchronous primitive is added to an internal work queue and may be executed on a different processor at some later time. Asynchronously scheduled primitives give the runtime more flexibility but incur more overhead, so it is beneficial to execute shorter primitives synchronously.

3.3.2 Phylanx Project Organization

The Phylanx project comprises three teams, each located at a different academic institution. The **Runtime Team** develops the HPX and Phylanx libraries. The **Performance Analysis Team** develops instrumentation to collect performance data and tools to improve performance. They also maintain the nightly regression tests and reporting. The **Visualization Team** develops visual tools to aid in performance analysis and debugging. Additionally, a program manager (PM) for the project seeks out further collaborations and develops data science applications using the Phylanx system. A list of team members involved in the design process and their roles is available in the supplemental material.

We discuss the inception and further organization of the project within the framework of the design study methodology of Seldmair et al. [133], specifically the *winnow* and *cast* phases. We note which pitfalls were accepted and what other aspects of the project helped mitigate the negative affects of those pitfalls.

Accepting Winnowing Pitfalls

Prior to the official project start, the Performance Analysis PI and the Visualization PI had several conversations regarding difficulties in analyzing ATRs. The Visualization PI had faced the issue of traditional data collection being insufficient to analyze ATRs [65]. The Performance Analysis PI expressed difficulty in making sense of the data that could be collected. He noted there was little point in spending development resources and overhead on data that could not be analyzed.

These two coupled issues, (1) no data to analyze and (2) no analysis with which to use to the data, present a "chicken and egg" barrier to improving understanding and performance of ATRs. The two PIs view determining what data to collect a research goal of the project.

The data being an evolving target of research, along with the development of Phylanx itself and its changing concerns, means the project runs afoul of two of Sedlmair et al.'s winnowing pitfalls:

PF-4: No Real Data Available (Yet). During the project, the structure of the data and the format of the data have been evolving. Other potential sources of data are not yet instrumented. It is difficult to arrive at a final visual solution without finalized data.

PF-10: No Real/Important/Recurring Task. The fact that the data is in flux means tasks involving that data are also in flux. Furthermore, as Phylanx is developing rapidly, the concerns of the team members change over time, affecting their higher-level goals.

The decision to proceed despite these pitfalls was motivated by the desire to solve the larger problem of performance optimization and analysis for ATRs. We view working with preliminary and in-flux data as a stepping stone to achieving the “data behind the data”—the data that can only be envisioned with knowledge gained from exploring what we already know.

There are several factors that help with the continuing success of the project, despite the pitfalls:

Identification and availability of meaningful preliminary data. Though the data collection is its own area of research, the PIs foresaw the importance of the execution graph based on prior work and could confidently predict it would continue to be useful to understand. We hypothesized that some tasks would therefore remain stable (see Section 3.4). Furthermore, design did not begin until a preliminary dataset could be generated.

Strong interpersonal relationships. The cohesion of the three groups facilitated adaptation to new data. Teams were quick to clarify or explain changes in format and to react to requests to change. The trust among the teams allowed the Visualization Team to plan for future functionality with relatively low risk.

Overarching goal of the project did not change. The high level goals of performance analysis and optimization, along with the goal of discovering what data to collect, remained the same, though strategies employed by the users changed. Thus, high-level goals that are aided by visualization, such as understanding the execution, remained fixed.

Visualization considered a deliverable by entire project. All project teams recognize the visualization component as an outcome. Progress on the visualization is reported at the weekly full-project teleconference and included in all reports. The success of the project includes the success of the visualizations.

The incorporation of visualization as a project-wide outcome underscores the continuing approval and enthusiasm communicated by project gatekeepers, placing them in the High Power-High Interest quadrant of the matrix proposed by Crisan et al. [33]. Team members were not only authorized to spend time on the visualization, but encouraged to do so. We further discuss project roles below.

Casting Roles: Gatekeepers, Analysts, Experts

The Runtime and Performance Analysis PIs, project manager, and program manager all serve in the gatekeeper role, with the Runtime PI and project manager being the most central in allocating time with front-line analysts. One student was identified as a front-line analyst early in the project. As the project evolved, several other students with differing concerns (See Section 3.6) were cast in the role.

The gatekeepers also acted as front-line analysts. The PIs had similar technical goals as the students. The project and program managers were more representative of a second goal—communicating the project to outsiders. All gave feedback regarding designs throughout the project.

An interesting facet of the project is that almost every person is a form of tool builder. Sedlmair et al. noted the pitfall of mistaking fellow tool builders for front-line analysts. Here they are both because a major goal of the visualization is to help the tool builders in building their tools. Their role as tool builders further helped them accept working with an in-development visualization (See Section 3.7).

Some studies have found success in blurring the boundaries between domain and visualization experts [157]. Our project naturally maintained them, further avoiding the pitfalls of working with fellow tool builders [9]. We found communicating with mock ups and screen shots was sufficient—users did not need to learn the language of visualization. Furthermore, as the other teams trusted in the visualization expertise of the designers, they accepted change in the design over time.

3.4 Task analyses

We had three objectives in designing our visualization. We wanted to (1) support the analysis needs of our collaborators, (2) refine data collection and analysis for tasking models, and (3) prepare for future needs given the refined data collection and the progress of the Phylanx project. Through our multi-year collaboration, we assessed needs through general project meetings, focused visualization and performance analysis meetings, and informal interviews. From these, we developed a goal-to-task lattice (Fig. 3.2), which we updated as needs shifted. We elaborate on this process and present the lattice below.

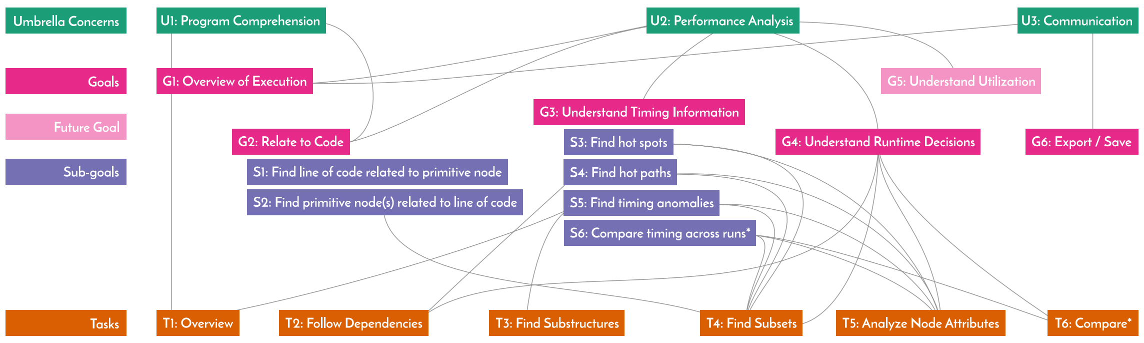

Umbrella Concerns
Goals
Future Goal
Sub-goals
Tasks
U1: Program Comprehension
U2: Performance Analysis
U3: Communication
G1: Overview of Execution
G2: Relate to Code
G3: Understand Timing Information
G4: Understand Runtime Decisions
G5: Understand Utilization
G6: Export / Save
S1: Find line of code related to primitive node
S2: Find primitive node(s) related to line of code
S3: Find hot spots
S4: Find hot paths
S5: Find timing anomalies
S6: Compare timing across runs*
T1: Overview
T2: Follow Dependencies
T3: Find Substructures
T4: Find Subsets
T5: Analyze Node Attributes
T6: Compare*

Figure 3.2: A goal-to-task lattice, showing the relationships between high-level umbrella concerns (U1 - U3); more specific goals (G1 - G6) and sub-goals (S1 - S6); and low-level tasks (T1 - T6) that directly inform the design of a visualization interface. The comparison sub-goal and task were added as data and concerns evolved. G5 was identified as a future goal based on project priorities in collecting and analyzing the data.

The project had weekly status meetings where all teams gave updates on the progress of individual components. Emerging problems were briefly discussed, but scheduled for another meeting if necessary. There was an optional meeting slot to discuss performance analysis and visualization specifically when requested. We wrote notes from both these meetings, including subjects not directly related to the visualization. We also had face-to-face meetings twice a year, once at another team's site and another at a conference in high performance computing.

Through the present, we created 152 note files with a mean 2800 characters per file. Some contractual information prevents us from releasing the complete audit trail [23] at this time, but anonymized summaries of our task analyses, with relationships between specific note files and the goal-to-task lattice are included as supplemental material. The project manager also compiled regular notes from the perspective of the Runtime team which augmented our understanding of the full project status and aided in our planning.

Tasks regarding the execution graph were derived from the note files by two authors independently who then developed a lattice spanning from high level goals to low level tasks using affinity diagramming. We classify these as *umbrella concerns* (U1 - U3), *goals* (G1 - G6), *sub-goals* (S1 - S6), and *tasks* (T1 - T6).

3.4.1 Umbrella Concerns

We use the term *umbrella concerns* to describe the major classes of goals we found our users had with respect to visual analysis. Some goals fell under multiple umbrella concerns.

U1. Program Comprehension. Our collaborators want to understand what happened when the program was executed. Many were working on a specific piece of the Phylanx pipeline and did not have a concrete mental model of how the translation from code to execution graph took place, nor a sense of the intermediate PhySL representation. Although previous work found that computing researchers may consult graphs to debug their mental model [37], we found some of our collaborators wanted to build their mental model. This is often a first step to devising new strategies, debugging, or performance analysis.

U2. Performance Analysis. An impetus for moving to tasking runtimes is the potential for high performance—decreasing the time to solutions and/or making previously infeasible computations feasible. Thus, understanding and improving the performance of a given Phylanx application or the system itself was a driving concern.

U3. Communication. Our collaborators wanted to create figures to help explain their own research in publications. The project and program managers were interested in explaining to potential users how the Phylanx system works. Such users often already have a background in parallel computing and thus can interpret the visualization when presented by someone from the team.

3.4.2 Goal-Task Lattice

We identified six goals relating to the execution graph and our umbrella concerns, some of which could be divided into smaller sub-goals. We discuss each goal and relate it to low-level tasks. We then summarize the tasks pulled from our goals.

G1. Overview of Execution. All three umbrella concerns wanted some sort of overview of what happened during the execution, in particular, the size and shape of the execution graph and how many times each node was executed. This goal can be divided into tasks of gaining a graph overview (T1), following dependencies (T2), and finding substructures (T3). For example, our collaborators explained that the visualization should allow them to understand if something was called recursively. This can be done by following a cycle of dependencies in the aggregated execution graph.

G2. Relate to Code. While the execution graph describes how the runtime executes the program, our collaborators cannot directly change the graph itself, only the associated source. Thus, they want to know the relationship between the code and the graph for both program comprehension and performance analysis concerns. We divide this into two sub-goals: (1) finding the line of code related to a node in the graph and (2) finding the nodes in the graph related to a line of code. The latter we categorize as a task of finding a subset of nodes (T4).

G3. Understanding Timing Information. Central to the Performance Analysis concern is data recorded about time spent executing each node. Of particular interest is finding parts of the execution that took a long time or behaved in an unexpected way, leading us to identify sub-goals of: (1) finding hot spots, (2) finding hot paths, and (3) finding timing anomalies. Hot spots are nodes that executed for a long time. Hot paths are sequences of such nodes.

Later in the project, as our collaborators progressed from the initial development of their applications to performance optimization, a fourth sub-goal, (4) comparing performance between runs, was discovered. We added it when we revisited our goal-task lattice.

All of these sub-goals require finding a subset of interesting nodes (T4) and analyzing attribute data of those nodes (T5). The hot paths sub-goal also requires following dependencies (T2). The timing anomalies sub-goal may further involve identifying substructures in the graph (T3) and understanding an overview (T1).

The comparison sub-goal requires comparing attribute data (T6).

G4. Understand Runtime Decisions. A key feature of tasking runtimes is built-in support for adaptively altering execution based on runtime data to improve performance. Our collaborators want to know what choices were made and the effect on performance, making this a Performance concern. An example of this goal is the choice of execution mode as described in Section 3.3.1. Similar to G3, understanding runtime decisions is aided by finding a subset of interesting nodes (T4) and analyzing node attribute data (about runtime parameters) (T5). As decisions are related to dependencies, following dependencies (T2) is another task. Also like G3, we updated the tasks for this goal with comparison (T6) as the objectives of our collaborators shifted.

G5. Understand Utilization. The primitives represented by the execution graph as nodes must be scheduled to run on computing resources. Researchers are interested in maximizing the utilization of those resources—spending less time idling and more time doing useful work. Thus, this was another Performance concern. However, neither examining this data nor the capability to associate utilization data with the execution graph was a development priority for our collaborators over the other goals. We thus included it in our goal-task lattice as a possible future node, with references to notes on the matter so we may look back on them should utilization become a more pressing concern.

G6. Export/Save. Supporting the Communication concern, our collaborators requested a mechanism for exporting and saving the visualization.

From the goals, we collect six low-level tasks. We list them here followed by their relationship to the task taxonomy for graphs of Lee et al. [87].

T1. Overview (4.4 Overview)

T2. Follow Dependencies (4.1.1 Adjacency, 4.3.1 Follow Path)

T3. Find Substructures (5 Higher Level Tasks)

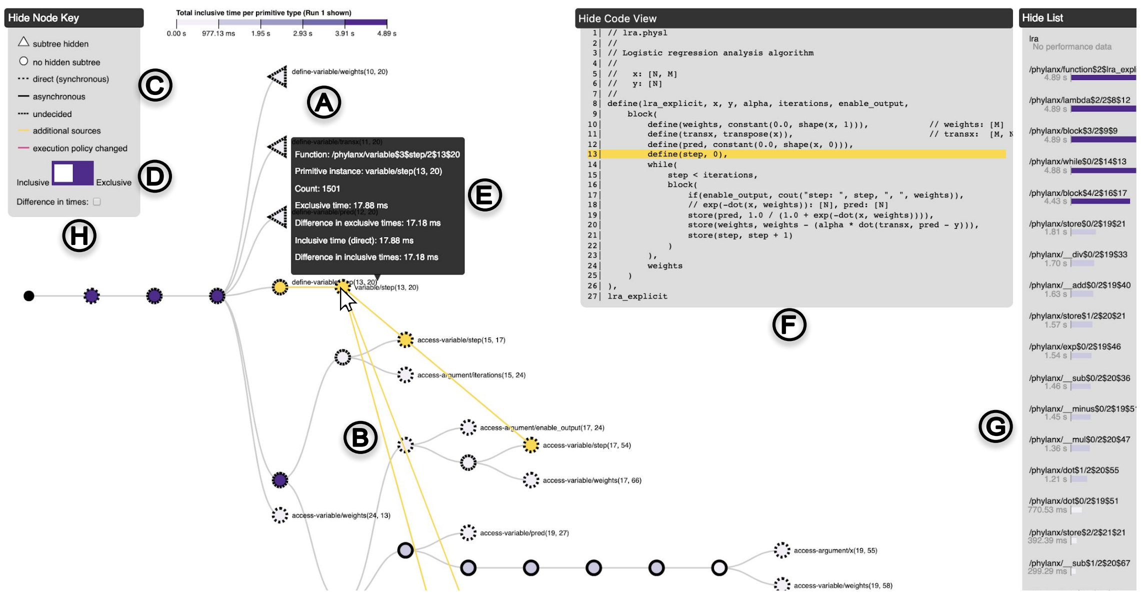
T4. Find Subsets (4.2.1 Node Attribute Tasks)

T5. Analyze Node Attributes (4.2.1 Node Attribute Tasks)

T6. Compare (4.2.1 Node Attribute Tasks, 5 Higher Level Tasks)

The presence of topology-based (finding adjacencies) and browsing (follow path) tasks when understanding dependencies motivates our use of visual representations that explicitly encode edges.

While there are several node attribute tasks, we note that we have relatively few attributes—timing data and mode of execution. This motivates our design decision to use on-node encoding, as it is easily understood by most users [111].


Hide Node Key
subtree hidden
no hidden subtree
direct (synchronous)
asynchronous
undecided
additional sources
execution policy changed
Inclusive
Exclusive
Difference in times:
Total inclusive time per primitive type (Run 1 shown)
0.00 s
977.13 ms
1.95 s
2.93 s
3.91 s
4.89 s
define-variable/weights(10, 20)
define-variable/transx(11, 20)
define-variable/pred(12, 20)
define-variable/step(13, 20)
variable/step(13, 20)
access-variable/step(15, 17)
access-argument/iterations(15, 24)
access-argument/enable_output(17, 24)
access-variable/step(17, 54)
access-variable/weights(17, 66)
access-variable/weights(24, 13)
access-variable/pred(19, 27)
access-argument/x(19, 55)
access-variable/weights(19, 58)
A
B
C
D
E
F
G
H
Hide Code View
Hide List

Function: /phylanx/variable3step/2$13$20
Primitive instance: variable/step(13, 20)
Count: 1501
Exclusive time: 17.88 ms
Difference in exclusive times: 17.18 ms
Inclusive time (direct): 17.88 ms
Difference in inclusive times: 17.18 ms

```
// lra.physl
//
// Logistic regression analysis algorithm
//
//   x: [N, M]
//   y: [N]
//
define(lra_explicit, x, y, alpha, iterations, enable_output,
    block(
        define(weights, constant(0.0, shape(x, 1))),           // weights: [M]
        define(transx, transpose(x)),                          // transx:  [M, N
        define(pred, constant(0.0, shape(x, 0))),
        define(step, 0),
        while(
            step < iterations,
            block(
                if(enable_output, cout("step: ", step, ", ", weights)),
                // exp(-dot(x, weights)): [N], pred: [N]
                store(pred, 1.0 / (1.0 + exp(-dot(x, weights)))),
                store(weights, weights - (alpha * dot(transx, pred - y))),
                store(step, step + 1)
            )
        ),
        weights
    )
),
lra_explicit
```

```
lra
 No performance data
/phylanx/function$2$lra_expl
 4.89 s
/phylanx/lambda$2/2$8$12
 4.89 s
/phylanx/block$3/2$9$9
 4.89 s
/phylanx/while$0/2$14$13
 4.88 s
/phylanx/block$4/2$16$17
 4.43 s
/phylanx/store$0/2$19$21
 1.81 s
/phylanx/__div$0/2$19$33
 1.70 s
/phylanx/__add$0/2$19$40
 1.63 s
/phylanx/store$1/2$20$21
 1.57 s
/phylanx/exp$0/2$19$46
 1.54 s
/phylanx/__sub$0/2$20$36
 1.46 s
/phylanx/__minus$0/2$19$5
 1.45 s
/phylanx/__mul$0/2$20$47
 1.36 s
/phylanx/dot$1/2$20$55
 1.21 s
/phylanx/dot$0/2$19$51
770.53 ms
/phylanx/store$2/2$21$21
392.39 ms
/phylanx/__sub$1/2$20$67
299.29 ms
```

Figure 3.3: The design of Atria. The main view represents the expression tree contained in the execution graph. (A) Triangles represent collapsed subtrees. (B) Elided links are shown on hover. (C) Fill color and border style encode time and execution mode respectively. (D) Users can toggle between showing inclusive and exclusive time. (E) Tooltips provide details on hover. (F) Code view with linked line of code highlighting. (G) Primitives listed by execution time. (H) If multiple runs are available, comparative mode may be enabled.

Evolution of the Goal-Task Lattice

We remark that our task analysis remained stable through multiple revisions. Later notes tended to reinforce goals and sub-goals already in the lattice. This may be due to the central need for comprehension of the execution as a starting point for any other goal. We hypothesize this relative stability over time contributed to the success of the visualization, despite the evolution of the data and the shift in focus towards comparison.

3.5 Visualization Design

Atria (Fig. 3.3), was designed and developed iteratively as data became available. We describe our design choices and explain how the evolving data, tasks, and environment influenced our design decisions.

The central view of Atria is the execution graph, visualized as a node-link tree. We explain this choice along with the choice of attribute encodings. We then describe the auxiliary linked views.

Throughout its development, Atria has served several purposes: (1) an initial validity check on data generated, (2) a visual tool supporting our collaborators in their evolving tasks (Section 3.4), and (3) a platform for hypothesizing about what new data to collect to help with the analysis. In support of these concerns, deployment of a working version was a priority. Matching the evolution of project concerns, we strongly embraced the advice [133] of satisfying needs rather than optimizing them. We describe the effect of these deployments on design, including significant changes for Jupyter Notebooks (Section 3.5.4).

3.5.1 Execution Graph

An execution graph is a directed acyclic graph of tasks describing the dependencies that must be met before any task (*primitive*) can be executed (Section 3.3). Rather than show all edges in the graph, we display a subset of the edges and lay out the graph as a tree. Specifically, we represent the execution graph as an *expression tree*.

In an expression tree, each node is an operation and its children are its operands. In Atria's graph view, each node is a primitive, which may be a simple or complex operation, and each child is an operand to that primitive as described by the PhySL

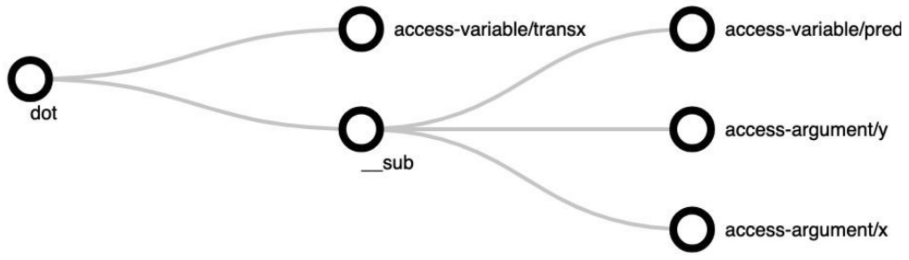


Figure 3.4: The expression tree of $\texttt{transx} \cdot (\texttt{pred} - \texttt{y} - \texttt{x})$.

intermediate representation. Fig. 3.4 shows a small example. We chose to prioritize expression tree links because of their relation to the PhySL code and to descriptions of the Phylanx model we had gathered from discussions with collaborators and their presentations.

Our collaborators' interest in the expression tree abstraction drove the evolution of the data collection. First, we collected only expression tree data. We created three interactive tree visualizations using icicle plots, node-link diagrams, and indented trees. Each allowed collapsing of sub-trees into single marks (triangles in our node-link tree). Within a few months, it became clear from viewing only the expression tree that there are cases where the execution graph is needed for analysis.

We created mock ups showing a full-graph node-link diagram as well as options for augmenting the tree visualizations with the extra edges. Our collaborators uniformly preferred the tree layout. We then decided to focus on the node-link representation (using D3's [16] Reingold-Tilford [120] layout) for the tree because that early visualization received the most use; node-link diagrams are already prevalent in the computing space [66]; and studies have shown the utility of node-link representations for path following tasks [50, 76] i.e., T2.

Mindful to avoid premature design commitment, we revisited the choice of tree representation later with a collaborator not involved earlier. He strongly preferred the node-link tree, saying "Whenever we were learning algorithms or something like that, we would draw it like that. It's more comfortable because we're more used to it and we can more easily see what's going on...Although the one in class might be drawn top down though." We chose the horizontal aspect ratio because most displays have more horizontal space.

This familiarity with node-link diagrams can further help users in identifying substructures (T3). The trade off is that node-link diagrams are not compact. Users had to zoom or scroll to gain an overview (T1).

Elided Structure and Interaction. By showing only the expression tree as dictated by the PhySL, we have removed two classes of edges: (1) dependencies between multiple accesses of the same variable, and (2) dependencies between multiple uses of the same function call. These are not usually needed for our users' goals. To support a more detailed analysis if necessary, we show them in a node-centric manner on demand. When a user hovers over a node, the edges are overlaid in yellow to be non-obtrusive.

Some sub-structures in the tree are common but rarely of interest to our collaborators. To de-clutter the visualization, we use two strategies. First, we automatically collapse sub-trees for a known set of "uninteresting" primitives. Second, for the on-demand links, we omit edges between library functions (e.g., `add`) as these do not communicate the structure of the application, but lower level information about the runtime that our collaborators do not expect to be of use.

Node Attribute Encodings. As timing and runtime decisions tie directly into our collaborators' goals (G3, G4), in particular task (T5), we encode timing and execution mode on-node. Execution time is encoded in the node's fill saturation, allowing users to find groups of nodes with similar timing in context (T3, T4). Execution mode is shown in the border's line style. The exact time and name of execution mode, as well as other attributes such as the execution count, are provided in a tooltip on node hover further supporting T5.

Users can switch between two concepts of execution time. Inclusive time is the wall-clock time taken for the primitive to execute. Exclusive time subtracts from the primitive any time that can be attributed to waiting on its children. Our collaborators are interested in both.

3.5.2 Auxiliary Views

A collapsible linked view displayed the PhySL code to support G2, relating to code. Sub-goals S1 and S2 are implemented as linked highlighting. The code view auto-scrolls on node hover. We experimented with showing the related C++ or Python, but PhySL was preferred.

To support S3, finding hot spots, we have a collapsible list view which shows the tree primitives from most time-consuming to least with colored bars matching the on-node time encoding. The view is similar in design to that of Intel's VTune, which is used by several of our collaborators. VTune works at a lower-level of abstraction and cannot list by primitive.

3.5.3 Designing for Comparison

During our design study, some of our collaborators began exploring the effect of adaptive policies that change execution modes at runtime. They make changes to these policies between runs and wish to compare the results. They reported opening multiple Atria instances. In response we added a comparison mode.

Discussions after we proposed a comparison view indicated that our collaborators only compare two runs at a time, allowing us to calculate a simple derived value. For timing comparison, we change the node fill from execution time to execution time *difference* using a diverging color scale. For execution mode comparison, we highlight (magenta) the borders of primitives that were executed differently between runs but keep the line-style encoding of the first dataset. These encoding changes support the discovery of interest subsets of nodes (T4) and their comparison (T6).

When only policies are changed, the structure of the tree does not change. However, when the application code or Phylanx changes, the tree structure will change. As Phylanx is under active development, we observed small topology changes every few weeks. When we observe nodes that are not present in both trees and thus cannot be compared, we draw them with lowered opacity, similar to the approach employed by Campello et al. [22]. So far topology comparison has not been a focus of our collaborators.

3.5.4 Design Changes for Deployment

Our primary Atria audience uses a web-based deployment. As the data collection, output, and use scenarios evolved, we created several design variants, resulting in multiple similar deployments, described in the supplemental material.

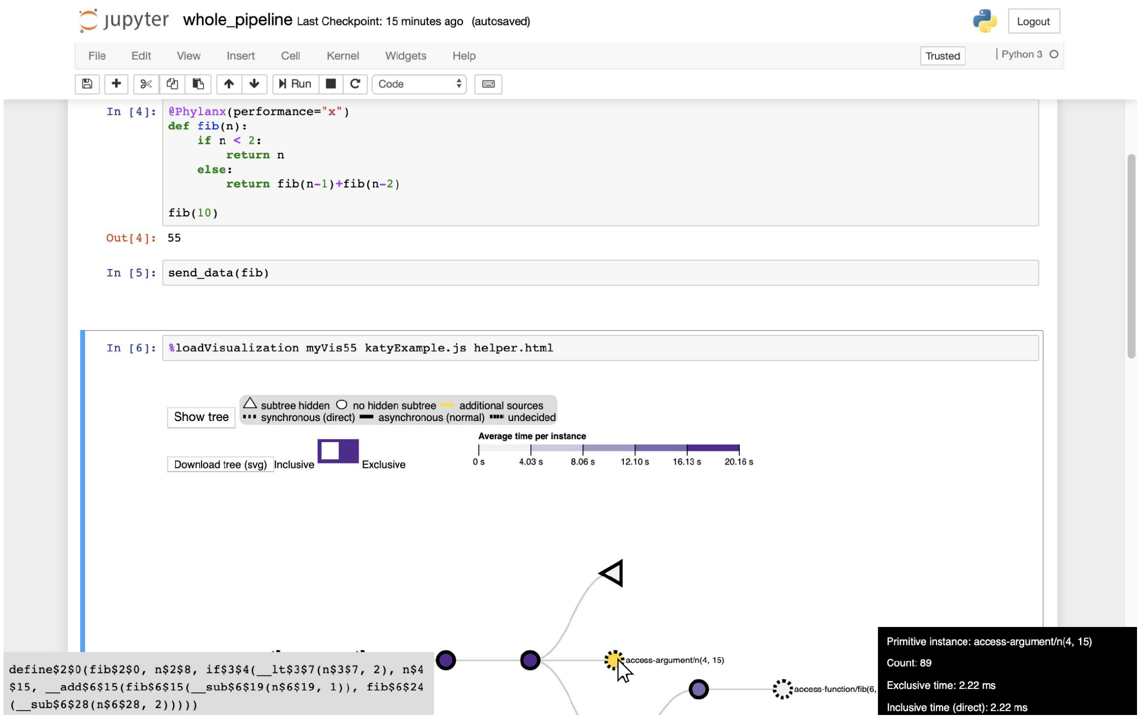
Output and collection changes, made by the Runtime team, were driven by visualization goals, specifically: (1) integration of Atria with automated nightly regression tests and (2) a full application-to-analysis demonstration in Jupyter, requested by team members with external communication goals (U3). We discuss changes for the latter below.

Atria in Jupyter

Jupyter Notebook is an interactive coding environment supporting literate programming. Users enter code into input cells that can be run (and re-run) to produce output cells. Variables persist through multiple input cells. Jupyter Notebooks are one of Phylanx's front-ends, available through Docker containers. The front-end is important to the project due to the ease and portability of container installation combined with the prevalence of Jupyter in the data science community.

The project and program managers give Phylanx demonstrations through this front-end and thus wanted Atria integration to help explain the programming model. Once we had an initial Jupyter pipeline in place, we found additional users who wanted to test out small code snippets and see the effects. An example notebook is shown in Fig. 3.5. The export/save (G6) functionality was prioritized as users wanted to further share results.

The Jupyter Notebook interface imposed an additional space constraint on Atria, decreasing the width to $\approx 60\%$ of the browser. Normally our users can devote an entire display to the visualization. We modified Atria's layout of auxiliary views to prioritize visibility of the graph. Tool tip data was moved to a fixed position in the



```
In [4]: @Phylanx(performance="x")
        def fib(n):
            if n < 2:
                return n
            else:
                return fib(n-1)+fib(n-2)

        fib(10)

Out[4]: 55

In [5]: send_data(fib)

In [6]: %loadVisualization myVis55 katyExample.js helper.html
```

```
define$2$0(fib$2$0, n$2$8, if$3$4(__lt$3$7(n$3$7, 2), n$4
$15, __add$6$15(fib$6$15(__sub$6$19(n$6$19, 1)), fib$6$24
(__sub$6$28(n$6$28, 2)))))
```

```
Primitive instance: access-argument/n(4, 15)
Count: 89
Exclusive time: 2.22 ms
Inclusive time (direct): 2.22 ms
```

Figure 3.5: Atria in Jupyter Notebook. Cell 4 is Python code that uses the Phylanx library. The newly-generated performance data and tree are passed in Cell 5. Atria is loaded and displayed in Cell 6, with the generated PhySL shown in the bottom left and the details of the hovered node shown in the bottom right.

bottom right so it did not obscure the graph or legend.

We decreased the size of the code view and placed it in a floating window in the bottom left. It shows three lines of code, which we determined was enough context for our users during formative evaluation. Users can output the full PhySL to a separate Jupyter cell, which was done during demonstration. The Jupyter interface itself thus acted as another (full) code view, available via scrolling.

Jupyter is hosted in a web environment with its own structure, styling, and handling of Javascript. This posed technical challenges in embedding our Javascript visualization in a cell in a maintainable manner. We view streamlining of this process as an avenue for future work.

3.6 Evaluation

We evaluate Atria and its inclusion in the Phylanx project through case studies gathered during deployment and evaluation sessions. Additional figures describing the evaluation and a video showing the case study of Section 3.6.1 are included as supplementary material.

3.6.1 Deployment Case Studies

As described in Section 3.5.4, we prioritized deploying versions to our collaborators, creating several variants during the project. Additionally, the program manager created a variant for his workflow at a secure facility. Data collection and design streamlining done for earlier deployments made this last deployment possible.

We polled our collaborators for their non-evaluative uses of Atria every few months. R3 consistently reported using Atria as described below. One other student reported using it actively when they were working on a particular algorithm, but has since changed objectives and does not presently use it. The program manager reported using it sporadically to explain the project to others. In the evaluation sessions (Section 3.6.2), four participants report using it minimally.

We describe two case studies. The first shows how Atria is used regularly in

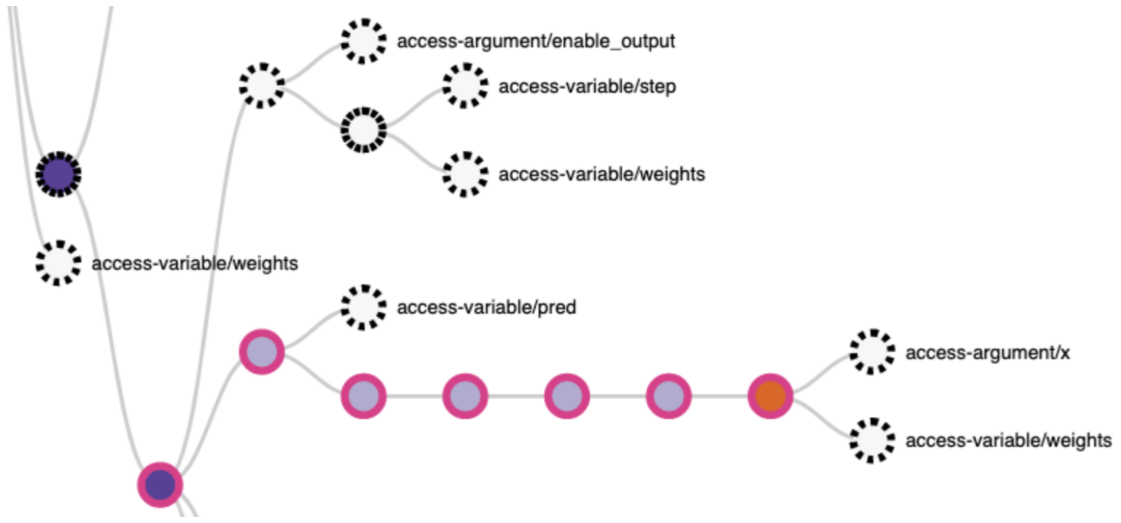


Figure 3.6: Comparison between two runs of the same application with different policies. Pink-outlined nodes indicate a difference in execution mode between two runs. The orange node ran slower after the policy change, but the net affect on the parents was positive.

Phylanx development. The second describes how Atria was used in a reactive situation to aid in reasoning and how the Atria development process influenced the performance debugging process.

Atria in Regular Use

Our primary frontline analyst, R3, began using the deployed version of Atria within a few months of the start of design in January 2018. He reported using the visualization on average once a week, more frequently when actively debugging.

He first runs the application he wishes to examine, generating the data used by Atria. He copies the files to a local directory and opens Atria from the command line. He considers the overall shape of the tree, noting that nodes with similar depth may be candidates to run concurrently. Then, he considers a particular primitive and its children to examine how the timing and execution of the children may have affected the parent as shown in Fig. 3.6.

Using his gained intuition, R3 makes a change in the Phylanx policy. He changes the thresholds that determine whether a primitive will be run synchronously or asynchronously. He runs the program with the new policy and collects data. Using Atria, he compares the two runs to see which of the primitives changed their execution policies and whether that caused them to run faster or slower. As the policy change is global and timing changes may have non-local scheduling effects, he browses the entire tree. He uses his findings to inform the next iteration of policy development.

When explaining his workflow to us, R3 said “Also it’s that I want to be able to visualize it [the algorithm], just seeing it implants it in my mind.” He explained that he is a visual person and Atria makes it easier to think about the problem.

Investigation of Performance Regression

A significant slowdown in Phylanx’s alternating least squares (ALS) application was discovered through nightly regression tests. The project manager (R2) suggested using Atria to compare runs before and after the performance drop.

The regression tests ran with a large and a small dataset. Atria data was collected only for the small run and showed no odd behavior, indicating further examination of the larger run was required. As a test dataset, the visualization team collected data using the older (pre-slowdown) code on a different cluster. No performance difference was observed, indicating the behavior was machine-specific.

The Performance Analysis PI (P1) then collected the larger run data on the regression machine. He discovered the problem was due to a change at the HPX level. He suggested it would therefore not be visible in Atria. The Runtime PI (R1) hypothesized it would show up as a 10-30% increase in all primitives on average. We used Atria to compare the two versions (Fig. 3.7) and found R1’s hypothesis to be correct.

Atria did not pinpoint the source of the problem, but was used to narrow the space of possibilities and then confirm understanding of lower level effects on the application. Furthermore, the involvement of Atria motivated deeper examination of the problem and the data collection that led to discovery of the root cause which was then fixed.

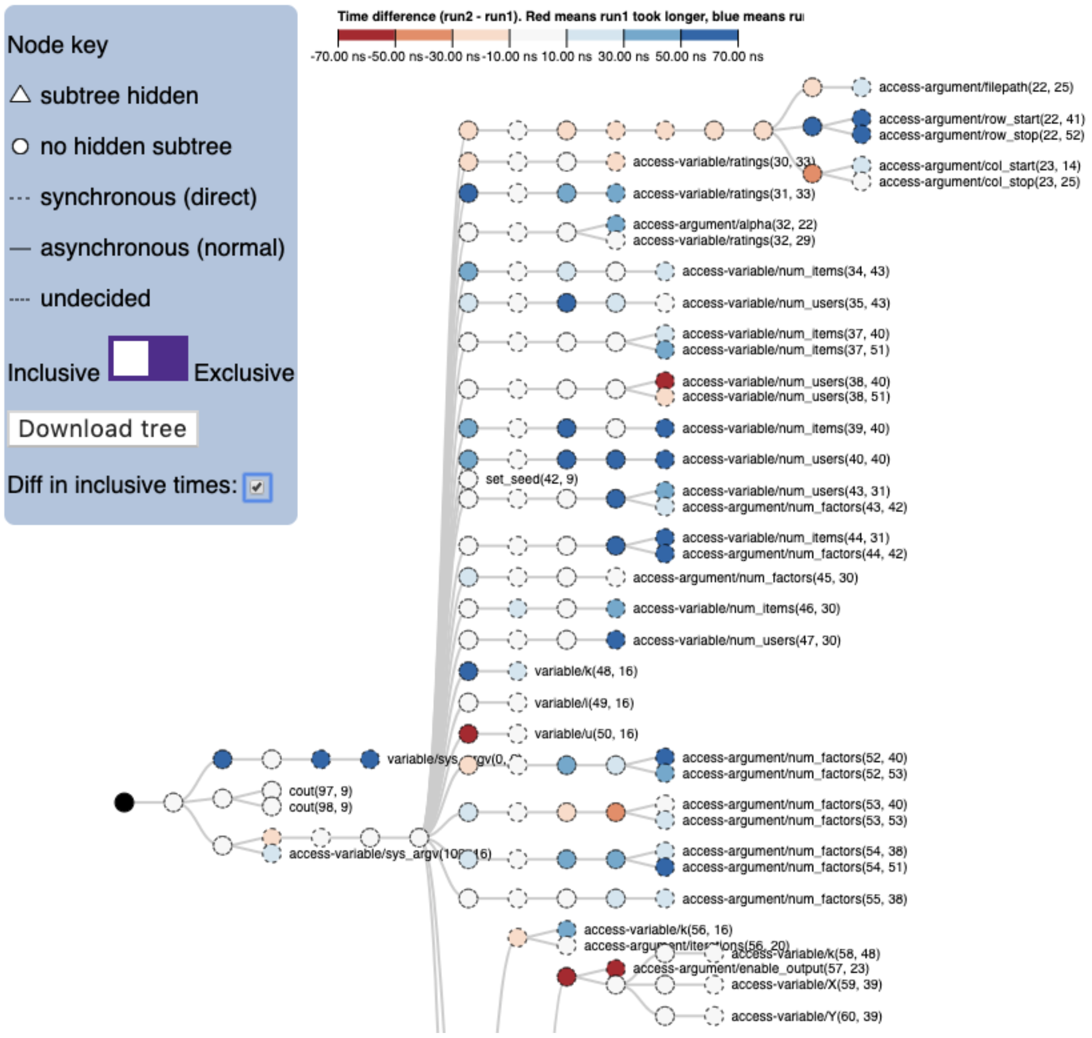

Node key
△ subtree hidden
○ no hidden subtree
synchronous (direct)
asynchronous (normal)
undecided
Inclusive
Exclusive
Download tree
Diff in inclusive times:
Time difference (run2 - run1). Red means run1 took longer, blue means ru
-70.00 ns -50.00 ns -30.00 ns -10.00 ns 10.00 ns 30.00 ns 50.00 ns 70.00 ns
access-argument/filepath(22, 25)
access-argument/row_start(22, 41)
access-argument/row_stop(22, 52)
access-argument/col_start(23, 14)
access-argument/col_stop(23, 25)
access-variable/ratings(30, 33)
access-variable/ratings(31, 33)
access-argument/alpha(32, 22)
access-variable/ratings(32, 29)
access-variable/num_items(34, 43)
access-variable/num_users(35, 43)
access-variable/num_items(37, 40)
access-variable/num_items(37, 51)
access-variable/num_users(38, 40)
access-variable/num_users(38, 51)
access-variable/num_items(39, 40)
access-variable/num_users(40, 40)
set_seed(42, 9)
access-variable/num_users(43, 31)
access-argument/num_factors(43, 42)
access-variable/num_items(44, 31)
access-argument/num_factors(44, 42)
access-argument/num_factors(45, 30)
access-variable/num_items(46, 30)
access-variable/num_users(47, 30)
variable/k(48, 16)
variable/i(49, 16)
variable/u(50, 16)
variable/sys_argv(0,
access-argument/num_factors(52, 40)
access-argument/num_factors(52, 53)
cout(97, 9)
cout(98, 9)
access-argument/num_factors(53, 40)
access-argument/num_factors(53, 53)
access-variable/sys_argv(10, 16)
access-argument/num_factors(54, 38)
access-argument/num_factors(54, 51)
access-argument/num_factors(55, 38)
access-variable/k(56, 16)
access-argument/iterations(56, 20)
access-variable/k(58, 48)
access-argument/enable_output(57, 23)
access-variable/X(59, 39)
access-variable/Y(60, 39)

Figure 3.7: Comparison between two ALS runs, before and after a significant slowdown. As hypothesized by the Runtime PI, there would be a slight increase in execution time on average for the slower run (blue).

3.6.2 Evaluation Sessions

We conducted evaluation sessions of Atria with seven members of the Runtime Team (R4–R10). R10 had no prior experience. Some had seen Atria briefly (R5, R6, R8, R9), though R8 only remembered after completing the tasks and R9 had only seen a picture. Two (R4, R7) had previous influence on design. Sessions were conducted at the participant's workstation, with the exception of two (R6, R7) which were done in a nearby meeting area.

We began our evaluation sessions with a demonstration and feature overview using a small example. Participants could use Atria and ask questions. We then asked users to perform a series of tasks on a dataset generated from a Phylanx application with which they were not familiar. We followed up with a semi-structured interview and de-briefing.

Evaluation Sessions: Tasks

We asked the following evaluation tasks, each marked with its corresponding goal from Section 3.4. Tasks L1–L4 are for a lone run. Tasks C1–C2 are for comparative runs.

L1: Find a primitive that takes a lot of time. (G3)
  How long does it take without its children? With? (G3)

L2: Find a primitive that is executed synchronously (G4)

L3: Find a primitive that is executed asynchronously (G4)

L4: Find a primitive that is repeated in the code (G1)

C1: Which run was slower? (G3),
  *Why might it have been slower? (G1, G2, G3, G4)

C2: Find a primitive that changed execution mode. (G4)
  Explain the change. (G4)

Five of the participants were able to complete the L1 tasks within seconds, doing a visual search for the most saturated nodes and then reading the exact numbers from the tool tip. R5 attempted to use the list view, but it wasn't yet linked. They also required a reminder of the encoding before completion. R10 had difficulty and seemed to be still learning the visualization.

All participants were able to complete L2 within seconds. However, in L3, four the participants took tens of seconds. We believe this was because there was only one correct answer for L3 in the sample data. L4 was also completed by all participants, though two (R7, R10) asked for clarification.

In the comparison tasks, all participants answered which run took longer (C1), with many verbally reasoning about the colors. However, in two cases (R4, R5), the phrasing of the question accidentally included the follow-up hint. In finding the changed execution (C2), all participants with the exception of R8 completed the task, though two asked that the encoding be re-explained. While the other participants seemed to browse for a node, R8 flipped between the comparison and non-comparison mode, appearing to search for line style changes.

C1* was a higher level analysis task that required performance analysis background, thus we only asked it of participants who indicated they performed such analysis in their duties (R5–R7). Each pointed out highly saturated nodes of the slower run's color as contributing to the slowness and noted in particular a `store` primitive was among them. R7 suggested that since the `store` took a lot of time, the program might be memory-intensive. R5 noted he had outside knowledge—that the `store` primitive had been modified recently—and concluded the data might represent the performance change due to that modification.

Throughout the tasks, we noticed a few common themes in the interaction. In most cases, participants appeared to use the encodings and interactions as intended. A few verbalized their rationale. Several participants consulted the legend in solving tasks (R6, R8, R10). Difficulties arose in discerning the line borders (R4, R10) and in evaluating the last hovered node which remained yellow (R4, R5, R7).

Evaluation Sessions: Interviews

We conducted semi-structured interviews with participants, asking if there were any features they found useful and what they would like the visualization do that it could not already. If participants indicated they had used the visualization before, we asked them how they used it.

Regarding utility, two participants said they didn't know whether the features would be helpful or not (R6, R9). The remaining participants each listed several components of Atria, but there was little consensus among them. Repeated features included: access to timing data (P4, P5, P7), the linked code view (P4, P5, P8), the comparison view (P4, P5, P9), and links between dependencies (P5, P7, P8). Suggestions for improvement included differentiating primitive types (e.g., variables, functions, control-flow) (R6, R7) and more de-cluttering of the node-link tree (R6, R7).

Three participants (R4–R6) said they used a previous deployment to draw figures for a paper [140, 147] or report. R8 said they used it to view the structure of codes they were not familiar with and see timing data.

Evaluation Sessions: Discussion

In general, participants performed well on the evaluation session tasks with indications that our encodings were used to complete them. Most evaluation tasks were short as they were constrained by session length, but we consider them necessary to establish basic usability and to engage participants for the interviews.

The least experienced participant, R10, struggled with several tasks. R10 had just begun learning how to code. Our goal-task lattice focused on expert analysis or communication led by an expert, which may explain these observations.

The participants who performed the high level task, hypothesizing why one run was slower, showed a combination of reasoning including identifying hot spots (G3) and relating knowledge about code (G2).

The sessions also revealed confusion in the execution mode encoding. It was a recent change to support the new *undecided* mode. We attempted to preserve the previous encoding of solid and dashed borders with undecided being in between to match semantics. We are now reconsidering this choice. The confusion reveals a design challenge due to the shift in what we could assume about the data and a trade off with consistency with prior encodings.

We attempted to keep the interview question regarding useful features unbiased. We first asked if there are any features that were useful, stating that *no* and *I don't know* were helpful answers. Even still, we suspect participants were predisposed to answer positively. Though potentially biased, we were surprised by the variance in which features the participants found potentially useful.

We hypothesize this variance is due to the differing concerns of the participants. Some were starting to consider comparative performance analysis. Some were focused on development of specific Phylanx features or example applications. The two participants who answered they didn't know are focused on developing interfaces between existing libraries and Phylanx. They are not working on the execution of the task graph itself or on applications that decompose into the graph. The other participants do but in different contexts, perhaps leading to their difference in feature preferences.

3.7 Reflections and Lessons Learned

We discussed earlier (Section 3.3.2) the importance of the respect the Phylanx teams have for each other and the positioning of the data collection and visualization as goals of the project as a whole. These themes carry through several of the other lessons we learned throughout the project thus far. We discuss these lessons below.

When designing for a moving target, seeking to satisfy rather than optimize is essential. Recognizing that we were managing Sedlmair et al. [133] pitfall PF-10, *No Real/Important/Recurring Task*, we focused strongly on satisfying needs and deploying. We took a "wait and see" approach with optimizing particular encoding choices and functionality, not wanting to expend effort on features that might not last. This mindset also helped avoid pitfall PF-20, *Premature Design Commitment* as we espoused the changing nature of the data and project.

Similarly, our rapid deployments often contained UI bugs. These primarily decreased usability but did not change the meaning of the data—again satisfying rather than optimizing. We believe these were accepted by our collaborators because of the nature of active development throughout the whole project. The Visualization team reported runtime bugs to the Runtime team (opening tickets on Github), so the Runtime team naturally reported bugs with the visualization.

Task analysis and long-term corpus of notes help clamp down on reactivity. The design of Atria was part anticipatory and part reactionary. Both have risks. Anticipatory design may miss the mark. Reactionary design may support too short-lived a target. By grounding ourselves in a long history, we were able to judge any major addition in the context of long-term concerns.

Jupyter notebooks impose additional design constraints, but open a wealth of interaction opportunities. Of our deployments, Jupyter has required the most design changes and we foresee more as its use increases. While the space constraints are greater, the cell-based interface could augment or change how we develop and integrate interactive visualizations for use in this data science space. Our current design depends on non-visualization cells and scrolling to match functionality with our web version. This has worked well so far, but further development of design guidelines for interactive notebook environments are needed.

Rapid changes combined with multiple deployment targets incur a maintenance burden. While multiple deployments gave us many potential users and their diverse viewpoints, they imposed a development burden on the Visualization team. Each deployment is in a separate `git` branch and requires some manual effort when applying changes. We plan to delve further into how we can organize our code and development practices to decrease this burden.

Both the visualization and the design study process aided our collaborators in accomplishing their goals and helped establish a culture of data review. It can be difficult to discern whether it was a particular visualization that led to an insight, or the fact that anyone was looking at the data at all, especially in studies where visualization was not already in the domain experts' workflow.

The integration of the Visualization team and the design process made data collection and review a central priority. The dialogue between the teams and the rapid response to data exposed data collection bugs or mis-assumptions early. As seen in our regression case study (Section 3.6.1), the intervention of the design process worked in tandem with a specific analysis problem to reach a solution.

We recommend further examination of the benefits of visualization as an intervention, particularly with respect to developing best practices surrounding a culture of data review. Based on our experience, we attribute our success to the project organization from both sides. The rest of the project viewed the Visualization aspects as first class deliverables. In turn, as key members of the project, the Visualization team was also fully invested in other aspects of the project. Although this investment brings certain risks, its rewards include deep insights and impacts that are otherwise unavailable.

3.8 Conclusion

We presented a design study amid potential pitfalls regarding lack of data availability or task recurrence. The visualization outcome and the insights it supports have not been the only benefit to our domain collaborators. The design process itself and the integration with the visualization efforts have been beneficial, especially as an avenue for refining data collection and analysis practices. Instead of running into a data design problem, we successfully avoided rebuilding the visualization by discussing the underlying data abstraction and our users' mental models. This discussion strengthened our visualization design and clarified the goals of our data collection.

One of the goals of the collaboration is to research what data needs to be collected for asynchronous tasking runtimes. The evolution of the data has been in response to intuition gained in analysis. The process has also resulted in rapid verification of collected data and insights into current problems that we anticipate will form a strong foundation for the on-going, long-term design study.

Although we accepted some pitfalls as part of the project, several factors aided us in managing them. Project organization was a large factor—teams had respect for each others' expertise, met regularly, valued each others' time and deadlines, and viewed the contributions of all teams as project deliverables. The high level of participation resulted in a large corpus of design data collected both by the Visualization team and the Runtime team. This documentation was revisited frequently, both formally, through revising the task analysis, and informally, to guide design efforts, avoiding ephemeral needs. Acknowledging that the data and tasks were in flux, our technology probe, Atria, satisfied those needs while keeping the focus on learning what data and tasks supported analysis rather than finalizing a tool design.

Family and Friends Summary: Our collaborators were in the midst of building a software tool to speed up the time it takes to run a computer program, and they wanted a visualization tool to help them see how well this tool was working. They wanted to visualize the duration of individual parts of the program, while also being able to understand how those individual parts related to each other. Sometimes program parts can be run at the same time (i.e., in *parallel*), but other times they depend on each other and need to be executed in order (i.e., *sequentially*). Our final visualization looks like a family tree (see Figure 3.3), where the parts that can be run in parallel look like side-by-side branches and the parts that have to be run sequentially are the circles within those branches. What makes this visualization unique is how we built it: we had to constantly change and tweak our design because the data kept changing. Our collaborators were still figuring out what data to collect to visualize while we tested various visualization ideas. All these changes led us to title the paper "Visualizing a Moving Target." Fortunately, the constant communication between our teams and the relatively stable visualization goal meant that we could build a useful visualization that was able to adapt as the project evolved.

CHAPTER 4

Guidelines for Pursuing and Revealing Data Abstractions

In the last chapter, my collaborators and I debated changing our visualization from a tree abstraction to a graph abstraction. Ultimately, to match our users' mental models, we kept the tree abstraction and allowed for the graph details to be shown on demand in the visualization. During our design discussions, we did not directly ask about the details of our users' mental models of computing programs; instead, details emerged in snippets of conversations, like "[The tree visualization is] more comfortable because we're more used to it and we can more easily see what's going on." Comments like these sparked separate conversations with Kate Isaacs and Alex Bigelow about the difficulties in effectively communicating with data workers about data abstractions. A *data abstraction* is an underlying structure of the data that affects how it can be visualized, e.g., spatial data may include geographic coordinates and can be visualized as a map (but cannot be visualized as a bar chart). There are different ways to visualize the same data abstraction (e.g., tabular data may be visualized as a bar chart or line chart). However, we wanted to investigate how easy it was for data workers to consider changing their dataset from one data abstraction to a different data abstraction. This transformation could lead to new insights into the data and increase the available visualization options.

As a result, we conducted a grounded theory investigation to explore how a diverse range of data workers across disciplines consider data abstractions. This investigation consisted of an online interactive survey in which participants were prompted to think of a dataset and choose the best fitting data abstraction from our typology: network/hierarchy, textual, grouped, spatial/temporal, media, or tabular (Figure 4.2). The participants also provided attributes of the data and examples of what this dataset looked like using the selected data abstraction. After detailing their dataset as one data abstraction, participants were asked to imagine their dataset as one of the other five data abstractions. For example, if Taylor chose "network/hierarchy" as the best-fitting data abstraction for their dataset, the survey would ask them to imagine their dataset as “spatial/temporal.” We hoped that the results of our survey would help us uncover a mapping or system of transformations between different abstractions in our abstraction typology.

We did not find such a mapping but through our grounded theory investigation (Figure 4.1), we found that undiscovered, unused latent data abstractions are common. We found that latent data abstractions have hidden implications on how we think about data. We defined a *latent data abstraction* to be a data abstraction that is meaningful and useful, yet undiscovered. It has yet to be fully elucidated, communicated, documented, and formatted. A data abstraction becomes less latent as coherent details are identified, as its details are spoken or written, and as its artifacts in a computer are actualized into relevant forms. For example, if you have a table of addresses in a city, insight might be gained by thinking about those addresses as a network, to see clusters of neighborhoods.

We also discovered these latent data abstractions were provocative. People had strong reactions to our terminology and data abstractions. They did not like changes in their mental models about their data, and these disruptions provoked discussion that helped clarify terminology and what exactly they envision in the data set. Introducing a data abstraction typology, a model that describes the space of possible data abstractions and/or data wrangling operations, sparked discussion and elicited more specific communication about the dataset and abstraction, even when the typology was imperfect.

This work was previously presented at IEEE VIS 2020 and the published work is available [13][1]. The content has been modified slightly to fit the flow of this dissertation. This work was done by Alex Bigelow, Katherine (Kate) Isaacs, and myself.

This work was supported by the United States Department of Defense through DTIC Contract FA8075-14-D-0002-0007, the National Science Foundation under NSF III-1656958 and NSF III-1844573, and UA Health Sciences through the Data Science Fellows program.

[1]`doi:10.1109/TVCG.2020.3030355`

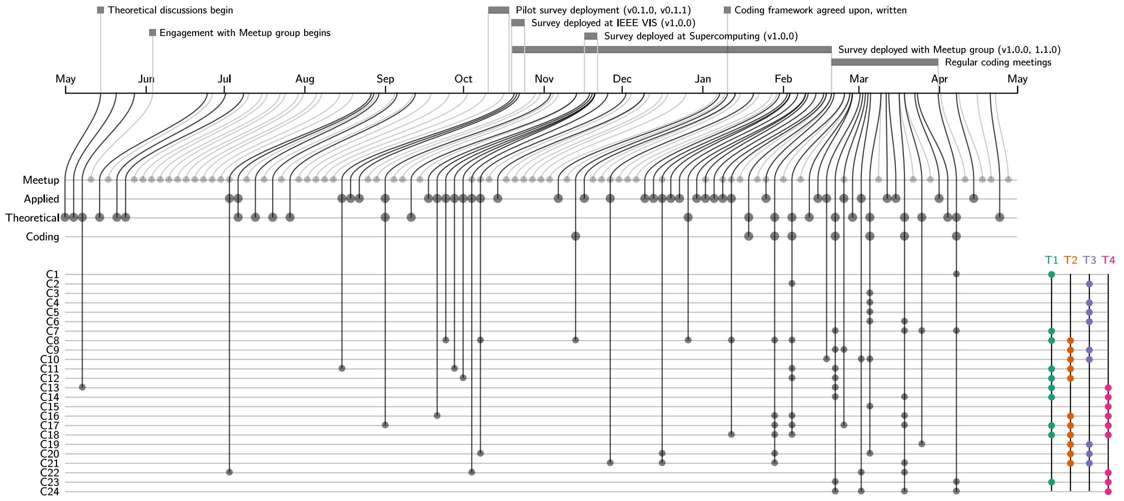


Figure 4.1: A summary of study events over time, their temporal relationship with memos, memo relationships with codes, and code relationships with themes. The timeline at the top shows the timing of study events, with curved lines indicating when individual memos were created. The four rows below the timeline indicate the nature of the context in which memos were written, including Meetup attendance, when data workers discussed their applied datasets, when the authors engaged in theoretical discussions, and when the authors engaged in open coding. Rows C1-C24 show which memos directly informed the development of codes. Columns **T1**–**T4** show which codes directly inform which themes.

4.1 Introduction

Data abstractions are fundamental to a wide set of visualization activities, from performing and documenting the provenance of data wrangling operations, to understanding the mental models of domain experts in design study research, to justifying design decisions in technique- or systems-focused research, and to reasoning about the role of data abstraction in theoretical visualization research. Difficulties in reasoning about and communicating data abstractions therefore have far-reaching implications: effective communication about data abstractions is critically important to the way researchers justify design decisions in technique- or systems-focused research. A poor understanding of the mental models of domain experts in design study research is a significant threat that risks creating solutions and systems that do not address real needs [103]. Too much focus on a single data abstraction has been observed to limit creativity [10] and to warp scientific analysis [8]. However, the extent to which these effects apply, in terms of specific abstractions, is poorly understood.

We set out to understand how malleable a data abstraction is, and to better understand the process of pursuing **latent data abstractions**. We define a latent data abstraction to be a data abstraction that is meaningful and useful, yet undiscovered. It has yet to be fully elucidated, communicated, documented, and formatted. A data abstraction becomes less latent as coherent details are identified, as its details are spoken or written, and as its artifacts in a computer are actualized into relevant forms.

Because there were blind spots in the questions that we should even ask, we chose to conduct a grounded theory method investigation seeking to discover how a diverse range of **data workers**, from spreadsheet users to programmers, across different disciplines, consider different data abstractions. This investigation analyzes **memos**, or research field notes taken during conversations, meetings, and interviews, as well as the results of a deployed survey.

The result is an evidence-based set of codes and themes regarding data abstractions with implications for how visualization project teams and individuals discover, wrangle, manage, and report their data abstractions. In particular, we find that introducing a **data abstraction typology**—a model that describes the space of possible data abstractions and/or data wrangling operations—can elicit rich communication and reflection about data and uncover latent data abstractions, even when such a typology is imperfect. We show how visualization researchers can increase actionable communication with data workers by introducing and critiquing a typology together, as a visualization design activity [95].

The codes and themes in this paper also add to existing literature by explaining some of the reasons why communicating about data abstractions can be so challenging. Reflecting on these themes and our collective interactions with data workers, we provide guidelines for communicating with data workers about data abstractions. These guidelines also have applications for more crisp communication about data abstractions in design study, technique, systems, and theoretical research papers.

We have further made the raw data collected in our survey available through an open interactive visual interface[2] along with an archive of codes, themes, an audit trail [24], and their revision history[3] that summarize observations recorded in memos from a year of interviews and meetings with diverse data workers, as well as observations from the visualized survey responses.

In summary, our contributions are:

1. A set of themes, supported by codes, that describe phenomena associated with data abstractions that arise in the processes of visualization design and data wrangling (section 4.5),
2. Guidelines regarding the development of data abstractions (subsection 4.6.1),
3. The design of an open survey regarding the description of data and the malleability of data abstractions (subsection 4.3.3, subsection 4.6.4), and

[2]osf.io archive of survey responses: `https://osf.io/s2wmp/`

Selected response visualizations are included in the supplemental material

[3]osf.io archive of codes, themes, and audit: `https://osf.io/382fn/`

4. An open dataset of survey results with a corresponding interactive visualization.

We begin by discussing necessary background and a review of related work (section 4.2), and our methodology (section 4.3). We present the codes derived from our study (section 4.4) and how they come together to form themes (section 4.5). We follow with guidelines and reflections (section 4.6).

4.2 Background and Related Work

We discuss the theoretical underpinnings of our work (subsection 4.2.1), related background in thinking and communicating about data in analysis and design projects (subsection 4.2.2), the importance of documenting real-world wrangling needs (subsection 4.2.3), and the context in which this work fits into research into creativity (subsection 4.2.4).

4.2.1 Theoretical Underpinnings

This study employs a team-based [154], interpretivist form [153] of grounded theory methodology, resulting in the development and refinement of four themes—these four themes, with their supporting codes, comprise what is often termed a *substantive theory* [101].

Interpretivist research has a different philosophical goal than the *positivist* research that we typically see in the visualization research community [97]. Interpretivist research aims to *describe* phenomena and *generate* hypotheses. This is in contrast to the positivist approach used in the scientific method that aims to *test* hypotheses. The four interpretivist themes that we identify, and their supporting codes, are transferable, in contrast to the way that formal theories are generalizable. Both intellectual traditions require systematic analysis of evidence, but the nature of supporting data and the ways that data are collected and analyzed are different.

Grounded theory methodology was an appropriate fit for beginning this investigation because our initial suspicions—that non-tabular data abstractions may be comprehensible, useful, and under-utilized among the broad population of data workers—were very general and based on a small number of surprising observations [8, 11]. The nature of the questions that we should pursue were prone to rapid revision and refinement as additional, surprising observations arose. Grounded theory is an approach that is uniquely suited for investigating and describing phenomena in which questions evolve rapidly, as themes are constructed by the data as it is being collected.

Consequently, we used *surprise* as a principled way to collect data [101]; the extent to which we pursued interactions with data workers, and adapted and deployed a survey, were motivated by identifying gaps in our own knowledge and unanticipated findings. In contrast, we also used our lack of surprise as a qualitative indicator to know when *codes*—or concepts that describe phenomena—had reached saturation, and needed no further investigation.

In presenting qualitative research, we are careful of pitfalls [127] in our reporting of numbers and counts: we include the full visualized corpus of survey responses[2] to maximize available context. Our numeric statements and visualizations are meant to be interpretivist descriptions of the phenomena associated with how data workers think and communicate about data abstractions, not positivist statements of statistical significance.

Although this is not a visualization design study, Meyer and Dykes' six categories for judging and reporting rigor [97] are relevant for the kind of interpretivist research that we present. This research is *informed* by our relevant prior research experiences; *reflexive* in our efforts to *constantly compare* [27] collected data and gaps in our understanding; *abundant* through the number of survey participants and diversity of interview and Meetup participants; *plausible* through documented connections from memos and survey responses, to codes, and to themes; *resonant* in that the themes have broad implications for how visualization research is conducted and reported; and *transparent* through the public release of the survey, its responses, and the revision history of the evolution of our codes, themes, and relevant metadata.

4.2.2 Thinking and Communicating About Data

We build on other efforts to understand how data workers think and communicate about data. From the beginning of our research, our main focus has been to expand understanding of one specific approach identified by Muller *et al.* [102]: how data workers approach the *design* of their data, as opposed to *discovery*, *capture*, *curation*, and *creation*.

Many authors have noted the designed nature of data abstractions [98], such as the handcrafted nature of many cybersecurity datasets [80]. Feinberg observes that the mere use of a dataset makes the user a designer of its abstraction [42], even if many data workers may not be aware of their inherent flexibility. Consequently, there is a need to learn to develop a “data vision” to exercise discretion and creativity in designing abstractions [112]. This is especially important in light of a data designer’s ethical responsibility to structure data effectively [35], as the design of what is measured and how it is stored can be overtly political acts [115].

The responsibility to design effective abstractions does not always fall upon data workers in isolation. In the context of visualization design studies that involve individuals with diverse roles and expertise, designing effective data abstractions [103] and communicating effectively about those abstractions as they evolve [133], are critical to the success of a project.

However, difficulties arise in effectively communicating about data abstractions [119]. There are myriad aspects to data abstractions in design projects, such as adapting to data changes, anticipating edge cases, understanding technical constraints, articulating data-dependent interactions, communicating data mappings, and preserving data mapping integrity across iterations [150]. These difficulties are consistent with reports of there being surprisingly little documentation about the design of abstractions [161, 162]. The lack of documentation makes human decisions invisible and threatens future analysis. In strictly machine-learning contexts, some authors have gone as far as suggesting that “de-emphasizing the need to understand algorithms and models” [113] may be an effective way to increase trust in model predictions. We show that the inverse is also true: that education and transparency can foster healthy skepticism of data models and abstractions, which can be important for fairness and provenance. We argue that transparency about data abstractions can be especially important for data wrangling and visualization, in which data workers need to "interact not only with the interface but *with the data*" [150].

To facilitate communication about a particular project's specific data abstraction, the visualization research community often relies extensively upon data abstraction typologies [104, 43, 28]. Currently, the main purposes of such typologies are to guide a researcher in the selection of appropriate visual encodings, and to support transferability across different design studies. However, aside from highly contextual design study research itself, there is little data that reveals the extent to which the visualization research community's typologies are compatible with data workers' perspectives and language, and, although the interventionist nature of design study research is known [93], the effects of introducing foreign data concepts have yet to be described in detail.

4.2.3 Data about Applied Wrangling Needs

Little applied data wrangling work has been published in the visualization community, even though novel algorithms, data structures, and infrastructure need to be implemented in ways that correctly address nuanced worker needs. Such efforts often consume the bulk of the labor involved in applied visualization research [53, 75, 102], and can include rich refinements in terms of task clarity and data location that advance science and constitute important visualization research contributions in their own right [133], yet, without also engineering a polished visualization system, such work has lacked clear publication venues.

The lack of such work leaves a major gap in needed visualization research. Although our work includes a qualitative dataset that only begins to fill this gap, the extent to which data workers use or even consider different data abstractions is still difficult to analyze or test, as data wrangling decisions are rarely documented in research or in practice [161]. When such decisions are documented in research, they typically only exist as justification for a visualization design; resulting in limited information about the data abstraction, its provenance, and important documentation about how and why it was reshaped.

It can consequently be difficult to justify technique-driven or systems-focused research into general-purpose data wrangling software systems [53, 75, 145, 57, 90, 135, 12], as such efforts often lack grounding in real user needs. Instead, they are forced to rely upon past researcher experience, scant hints about real-world data wrangling precedents that exist in design study literature, and speculation about how data workers might think and what operations they might find useful. This study, and future standalone publications that are focused on data transformations, can help to better inform the design of such systems.

4.2.4 Creativity and Creative Roles

Discovering a latent data abstraction can have powerful creative benefits, such as inspiring radical visual innovations [108, 94]. Although the work that we present has implications for visualization researchers and their interactions with the broader population of data workers, our primary objective is to compare and contrast sets of creative objectives that can be held by any kind of data worker—including visualization researchers themselves. Consequently, we identify the role of an **abstraction theorist** that seeks to discover useful latent data abstractions, and contrast that objective against the broad set of all other concerns that a data worker may need to consider, such as data wrangling, data ownership, workflow management, the design and implementation of visualizations, evaluation, and reporting on visualization research.

Contrasting these roles is similar in spirit to Von Oech's popularized "explorer, artist, judge, warrior" creative roles [146]: "theorist" and "worker" may refer to distinct individuals in a collaborative environment, such as a visualization researcher and domain expert, or they could refer to different priorities that a single individual is considering on their own. Therefore, we describe differences through a pragmatic lens, instead of attempting to analyze different populations' creative styles or cognition [136].

We add to precedents for pragmatic guidance for creativity in visualization design, including the design of creativity workshops [51, 77] and creativity exercises [95]—we propose the pursuit of latent data abstractions as an additional creativity exercise, specific to the design of data itself.

4.3 Methodology

The evidence upon which we base our findings comes from two sources: memos and a deployed survey about data abstraction perspectives. It is important to note that, consistent with our interpretivist objectives, many of the following methods are deliberately *uncontrolled*—rather than testing hypotheses, our goal is to ask better questions. Here we discuss both sources of data, and the way that they both influenced, and were influenced by, our internal data abstraction typology.

4.3.1 Memos and Timeline

We wrote memos in four contexts: 1) regular attendance at data-centered community Meetups, 2) applied conversations with data workers in diverse contexts about their perspective on their data, 3) theoretical discussions about data abstractions among the authors, and 4) collaborative open coding sessions. A summary of all memos, their relationships with codes, and code relationships with themes, are shown in Figure 4.1, and an associated audit trail [24] is available in the supplemental material.[3]

This project began with theoretical conversations about the nature of data abstractions between the authors, that arose occasionally as part of regular meetings. Early on, we decided to engage with an existing local Meetup group that regularly met to seek or provide help with data: a core group of regular members met twice per week at a coffee shop or bar, and continued to meet remotely beginning in March due to social distancing measures. Members and visitors frequently brought laptops to show data and code that they were working with, to solicit advice or help with debugging in a casual context. The core group and its frequent visitors included a diverse array of researchers, administrators, and data scientists from the local university and surrounding community. As these meetings and interactions were largely *ad-hoc*, an accurate count of all potential informants is impossible to report, however, a selected subset of these community members—those that provided specific information that informed the development of a code—are shown in Table 4.1.

Later, as our survey was developed, it was deployed among this group, as well as at the IEEE VIS and Supercomputing conferences. Each of the 219 survey responses are included in the supplemental material.[2] Deployments of the survey often prompted conversations that provided additional valuable insight that we added to our growing set of memos.

As concepts and patterns began to be less surprising, the authors began to identify codes from supporting evidence, in a collaborative open coding environment similar to the one described by Wiener [154]. After writing and agreeing upon a framework for documenting codes in a version-controlled repository[3], the authors began to meet 2-3 times per week to discuss, refine, and write codes that we had identified as we reviewed survey responses and our individual field notes. As we discussed different patterns in the data, each author actively cited [100] supporting personal experience, memos from a related interview, or specific survey responses to support or contest the proposed code. Where personal experience was identified as evidence, additional memos were taken. As we began to observe broader themes across codes, these were also written, discussed, refined, and connected to codes. Finally, an audit was conducted to verify the nature of the source data, the relationships between memos and survey responses to codes, and the relationships between codes and themes; the result of the audit is visualized in Figure 4.1.

4.3.2 Data Abstraction Typology Evolution

We began our investigation by adapting the data abstraction typology described by Tamara Munzner [104] to a data wrangling context: our initial objective was to describe a design space of possible data wrangling operations, so we modeled operations as edges in a complete graph, connecting each of five broad data abstraction types, as shown on the left in Figure 4.2. Each edge represents a transition from one data type to another; for example, network modeling tools [57, 90, 135, 12] would largely support operations along an edge from "Tables" to "Networks & Trees." Self-edges describe wrangling operations that transform a dataset to a different form of the same type, such as transposing the rows and columns of a table.

We quickly discovered many weaknesses of this model, through our own theoretical discussions and applied conversations with data workers. Most datasets

Table 4.1: Informants

Informant	Role	Domain
I1	Professor	Medicine / Bioengineering
I2	Research Assistant	Linguistics
I3	Postdoctoral Researcher	Biology
I4	Research Director	Information technology
I5	Professor	Mathematics
I6	Research Director	Interdisciplinary institute
I7	Professor	Interdisciplinary institute
I8	Postdoctoral Researcher	Information science
I9	Postdoctoral Researcher	Biology
I10	Postdoctoral Researcher	Biology
I11	Program Coordinator	Interdisciplinary institute
I12	Postdoctoral Researcher	Bioinformatics
I13	Professor	Public health
I14	Professor	Biology
I15	Professor	Computer Science
I16	Professor	Computer Science
I17	Research Assistant	Computer Science
I18	Engineer	Industry
I19	Data Scientist	Industry

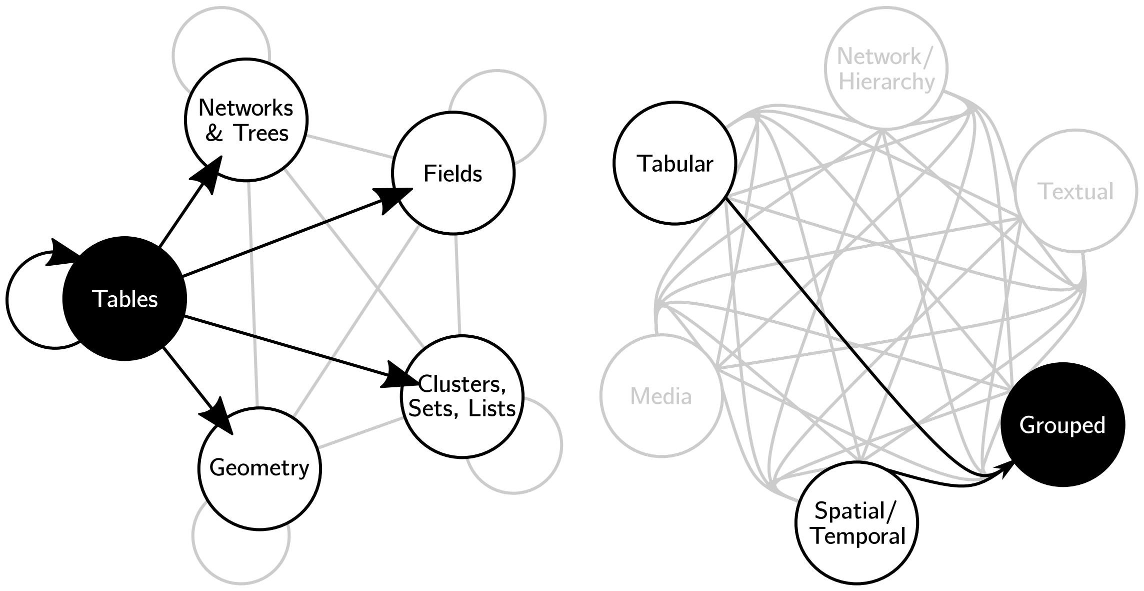


Figure 4.2: The evolution of our data abstraction typology. Initially, we modeled abstractions as fitting into five specific data abstraction types, with every node in the complete graph representing a potential latent abstraction (left). Data wrangling operations, such as converting rows in a table as nodes in a network, or performing dimensionality reduction of tabular columns in a high-dimensional space, are modeled as directed edges that require changing to a non-tabular data abstraction. As we engaged in applied conversations with data workers and designed our survey, the specific categories in our typology evolved, as did its model. The final typology models the process of considering a latent abstraction as a hyperedge coming from a hybrid set of different categories to a new target latent abstraction (right), such as imagining ways to cluster rows in a table based on columns containing geographic information.

have elements or relevant metadata that could be described heterogeneously: with more than one type of abstraction. Furthermore, many of the dataset types that we initially selected were a poor fit for specific datasets, such as text corpora. These weaknesses caused us to reflect on the overall purpose of such a model: one that attempts to delineate all possible data wrangling operations may not be possible. However, adapting it to be more flexible could potentially aid in the process of exploring latent data abstractions.

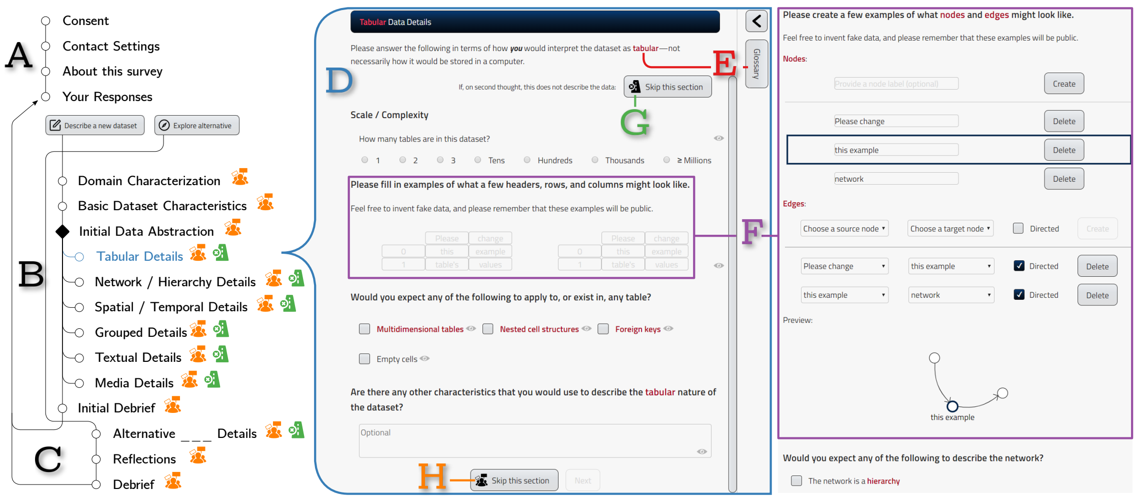
Motivated by the weaknesses that we had discovered about our initial model, we changed the focus of our investigation from modeling the space of possible data wrangling operations to investigating how malleable a set of identified abstraction types can be in practice, and the extent to which enforcing a different perspective on a real-world dataset can have creative benefits in its own right. We pivoted from attempting to develop a model, to developing and deploying a survey. Consequently, we do not present our preliminary data abstraction typology as a contribution, even though it guided the development of the survey and it informs our codes and themes.

For our survey, we adapted our model to describe the act of theorizing about an alternative data abstraction, instead of performing a concrete data wrangling operation—although it could still reveal unmet needs for data wrangling tools, that objective was no longer prioritized. We modeled these acts as hyperedges, also shown in the right side of Figure 4.2, with the target alternative abstraction remaining singular but allowing for any combination of source abstractions. This makes it possible for survey participants to describe their dataset with any combination of abstraction types, and yet still explore an abstraction type that may be less familiar.

4.3.3 Open-ended Survey Design and Deployment

We developed and deployed our survey in the form of an interactive web page. It is designed in three phases, shown in Figure 4.3: after the first introductory phase, the main phase of the survey invites the participant to describe a real or imagined dataset, in terms of our data abstraction typology's six broad data abstraction categories: tabular data, network / hierarchical data, spatial / temporal data, grouped data, textual data, or media. Further details are requested from participants where they indicate that they at least "rarely" interpret the data in terms of a particular abstraction category.

The final phase of the survey chooses randomly from the abstractions that a participant has indicated that they think about the *least*, and encourages them to try to think creatively about their data with that abstraction. It solicits qualitative, self-reported feedback on the extent to which the imagined transition is perceived to be useful, as well as which software tools participants would be likely to use to accomplish the transition.



Please answer the following in terms of how *you* would interpret the dataset as tabular—not necessarily how it would be stored in a computer.

If, on second thought, this does not describe the data:

How many tables are in this dataset?

Please fill in examples of what a few headers, rows, and columns might look like.

Feel free to invent fake data, and please remember that these examples will be public.

Would you expect any of the following to apply to, or exist in, any table?

Are there any other characteristics that you would use to describe the tabular nature of the dataset?

Please create a few examples of what nodes and edges might look like.

Feel free to invent fake data, and please remember that these examples will be public.

Would you expect any of the following to describe the network?

Figure 4.3: An overview of the survey that we deployed. The survey is divided into three sections, shown here as a flow diagram. The first section (A) includes consent forms, contact settings, an introduction to the innovations in the survey, and a summary of responses that redirect to the other two survey portions. The main "Describe a new dataset" portion of the survey (B) invites participants to describe a real or imagined dataset, and asks them to reflect upon the extent to which they think about the dataset in terms of the six dataset types that we identified. Where participants reply that they at least "rarely" think of their data in terms of a given type, they are asked for more details in a specialized Details section of the survey. The final "Explore alternative" portion of the survey (C) invites participants to imagine their dataset as the type that they initially thought about the *least*, and fill in the associated Details portion of the survey with this new perspective. As an example, the Tabular Details interface is shown (D). Participants are encouraged throughout the survey to look up terminology highlighted in red, where participants can edit the terms and suggest alternative definitions in the glossary (E). In some Details sections, participants are asked for a small sample of what they imagine the data to look like, to help ground their thinking (F). At any point in a Details section (G), or at the end of most other sections (H), participants can choose to skip the section to provide targeted critique on the survey itself if the questions have strayed far enough from the participant's mental model.

Throughout all phases, the survey solicits meta-feedback about the survey itself. Participants can challenge the survey design, questions asked, our set of possible data abstraction categories, and the terminology that we used. The glossary is interactive, allowing participants to provide alternate terms or definitions. Participants can also skip sections of the survey that begin to ask questions that the participant feels have strayed from their perspective or use case.

We deployed the survey among three groups: attendees at regular community data Meetups, attendees at the 2019 IEEE VIS Conference, and attendees at the 2019 Supercomputing Conference. Although the latter two groups had a less diverse computing focus, we were aware of ongoing discussions about data abstractions within these communities, and suspected that these groups were particularly likely to offer direct critiques of our typology and approach.

4.4 Codes

Here we present the codes for phenomena that we identified in our collaborative open coding process, with selected supporting evidence. For more detailed supporting evidence for each code, see the supplemental archive.[3] As there are 24 codes, we present them as groups for readability, however, the analyzed themes presented in section 4.5, and their relationships with codes, are more complex, as shown in Figure 4.1.

Codes **C1–C6** are based mostly on patterns that we observed in the visualized corpus of survey responses.

C1. Compared to the diverse responses in how participants described thinking about their data, the way that they characterized how it is represented in a computer was disproportionately tabular. This disconnect between the mental model and physical computer representation indicates not only a possible need for new data storage or data wrangling tools but also a lack of awareness of other data storage options. Data workers may default to tabular data organization because it more easily fits into their current workflow and tools, or because they do not know of existing "unconventional," non-tabular tools.

C2. There was wide variation in reported dataset scales. Taken from the median response for each of the "Basic Dataset Characteristics" questions (e.g.,"Approximately how large is this dataset?"), the median dataset was on the order of megabytes (close to gigabytes) in size, with thousands of items in the dataset and tens of attributes.

C3. Participants included broad techniques in their responses for wrangling tool support. When asked to actually transform their initial dataset into the alternative abstraction type, most participants listed software tools or programming languages but some listed techniques. These techniques included natural language processing ("NLP, Python", "Python, nlp techniques"), machine learning, and mathematical operations ("cluster into connected components", "Morse Smale Complex").

C4. Participants sometimes noted that they would need to ask a domain or visualization expert for help in order to change data abstractions. Along with techniques and software solutions appearing as answers to how the participant would actually transform the data abstraction, some participants acknowledged they either needed more information from a data theorist (e.g.,"Could be displayed as a tree, I would hire someone") or from a domain expert ("...would need to discuss this in more detail with a domain expert...this data was not provided").

C5. Participants sometimes noted that more information would need to be collected and added to the data before transitioning to a different abstraction. To transform their data from one abstraction to another, participants stated that they would need to collect additional data, such as images, speech transcripts, recordings, and labels.

C6. There was a wide distribution of the tools and techniques that data workers would use to wrangle data. Survey participants reported 54 different tools by name, with many tools being unique to a single participant. Tools that were mentioned by multiple participants tended to be programming languages.

Codes **C7–C12** are based on evidence from multiple sources, and are suggestive of unspoken perspectives, intuitions, and fears that may be common among data workers.

C7. Even before the survey guided participants to alternative abstrac-

tions, they discussed how they could see their data in other forms. This manifested both in conversations with participants before they took the survey, as well as in comments in the earliest sections of the survey before the question was asked.

C8. Many data workers did not feel that what they work with "counts as data." This comment was a common refrain while soliciting survey participation at both technical conferences, as well as through deployment across the university. However, outside of the survey, three informants (I1, I2, I4) independently made this observation while reflecting on their experiences working with people new to Data Science. For example, I2 often runs a data science workshop in the humanities but it tends to get very low attendance—often the same three participants. Seeing information as "data" may take a certain level of creativity and willingness to experiment and fail. One Supercomputing survey participant working on hardware design felt that treating circuit diagrams as "data" would be very strange, and perhaps inappropriate.

C9. Thinking about alternative data abstractions can provoke fears of scope creep. During a discussion with informants I6–I12, there was a consensus that exploring alternative abstractions can be very beneficial for the success of a project, however, it was also cautioned that it would have the potential to cause misalignments in the vision of a collaboration—usually termed "scope creep." Data workers are often cognizant of the impacts that changes to the design of their abstraction will have, including considerations and costs that they may or may not be able to articulate in detail.

C10. Data abstractions are often personal in nature to a data worker. Based on prior experiences, such as designing a visualization with I6–I12, the authors recognized that abstractions can be personal, subjective, and contextual. Wrapped in an existing data abstraction are a data worker's personal preferences, prior data science knowledge, and domain knowledge. Thus, suggestions to change this abstraction are often met with feelings of confusion and resistance. Some of these emotions stem from concerns about additional work overhead, such as those identified by **(C9)**. Other times, these emotions stem from the ecosystem of how the data was created, the people it may impact, and the subjects of the data—all things that a data worker may understand but a theorist may be unaware of.

C11. Data workers often have "gut feelings" or intuition about their data as networks. Data workers, regardless of whether their data is known to be network data or not, tended to have some intuition about the existence of networks within their data, even if specifics such as the meaning of a node or edge were unknown. Special types of networks, such as DAGs and trees, were also mentioned.

C12. Data workers often have "gut feelings" or intuition about their data as clusters, sets, or groups. Similar to **(C11)**, data workers also had intuition about the existence of groups in their data. They sometimes referred to hierarchies existing in and among these groups, and also intuited patterns and clusters in their data.

Codes **C13–C18** highlight informative weaknesses of our typology.

C13. There is wide variation in how data workers describe hierarchies. There was some initial difficulty designing the survey when deciding where hierarchies should fall. Even among the authors, we recognized that one could describe hierarchies as spatial, as networks, as nested sets. We questioned whether a tree and a hierarchy are the same thing, but concluded they have semantic differences. In the final survey, hierarchies were grouped with networks as a "Network/Hierarchy" abstraction type, with "Hierarchy" chosen deliberately to seek feedback. This diversity of perspectives was confirmed; one participant commented that they more closely align hierarchies with groups: "I find the separation of hierarchies and groupings to be a bit problematic for this domain. Many codes, such as diagnosis codes, exist in a hierarchy (defined by metadata). However it is quite common to refer to areas of this hierarchy as groupings."

C14. Most datasets did not fit in one category, and participants talked about not just the raw data, but derived values, metadata, or even "mul-

tiple datasets." Participants often selected multiple data abstractions in response to the initial question of categorizing their dataset. Heterogeneous datasets are very common, such as when metadata takes a different form from the main dataset, or when one dataset is a nested "value" inside another of a different type.

C15. "Media" as a category had a less well-defined mental model, resulting in a space with too little structure for participants to map their data crisply when forced to think of their data as "media." When asked to consider media as an alternative abstraction, a common response was to imagine screen-capturing to record images and video of a visualization of the data. But thinking of their data in this way elicited feelings of discomfort from some participants; comments such as: "This is weird. I think of the data not as media but I'm actively trying to turn it into media" and "I have displayed this data by mapping some of it [to color channels in a heatmap], but I don't consider the data itself to 'be' media or 'have' media." Some data workers understand some sort of inherent visual quality of their data. For example, one response was "The data set itself does not include any media, but interpretations of it are visual in nature... The data could be illustrated by addition of multidimensional images or 3D meshes when interlinked with concepts in the graph."

C16. Even very technical data workers find some data abstraction concepts, language foreign. We noticed confusion and misunderstanding surrounding our abstraction terminology; notably, terminology surrounding tabular data (e.g., items, attributes) was unknown to one Supercomputing participant and needed to be related to the physical spreadsheet (e.g., rows, columns) to clarify. This difference in theory-based thinking and practice-based thinking shows that there is a disconnect between how visualization people talk about data, and how data workers in general talk about data.

C17. Many data workers consider functions to be data. One unexpected finding, after reviewing responses aligning with **(C8)**, was that a subset of participants recognize functions as data. These datasets include continuous models, functions like regression models from housing data, collections of partial differential equations, or constraint data for linear or integer programming, which I5 and one author did not consider to be "spatial" as defined in the survey.

C18. Many data workers consider code to be data. As part of a larger discussion about open science and data sharing, several informants noted that code should be considered data. At a minimum, code acts as "metadata" by providing provenance of where a given dataset came from. As I6 noted that, "one person's metadata is another person's data."

Codes **C19–C21** describe the different ways that it was difficult to focus conversations with data workers on the design of a data abstraction.

C19. The design of a data abstraction proved difficult to talk about in isolation from specific file formats. Related to **(C16)**, some survey participants misunderstood the connection between an abstraction and its implementation (e.g. a table vs. a spreadsheet). As a result, in response to our request for "Other Generalizations," they suggested file formats that were clear fits for our existing six abstractions such as: "directed graph represented in a format such as dot" instead of Network/Hierarchy, "CSV file" instead of Tabular, "a collection of free text" instead of Textual.

C20. The design of data abstraction proved difficult to talk about in isolation from software and programming language abstractions. One author noted difficulties in focusing conversations on how a person thinks about their data; informants frequently pivoted to talking about abstractions imposed by software that were often only loosely associated with the data model itself, such as git's model of remotes and branches, or Jupyter's statefulness.

C21. The design of a data abstraction proved difficult to talk about in isolation from discovery, capture, curation, and creation. [101] Discussions often detoured from data design to topics such as data provenance and other data wrangling concerns. Similarly, when prompted to transform their data from one abstraction to another, some participants suggested collecting entirely new datasets, rather than transforming the existing data.

Codes **C22–C24** describe things that appeared to aid reflection and communication about data abstractions.

C22. Showing real data, such as a spreadsheet, helps data workers and theorists communicate effectively about data abstractions. Many different interactions at community meetups, such as with I3 and I19, were enhanced by the culture of bringing laptops to show data and inspect it together.

C23. Data abstraction typologies help data workers discover latent data abstractions. Asking questions about a data abstraction and how it fit, or did not fit, into a typology helped expand data workers' view of their dataset. One participant noted: "The questions made me think more about 'the nature' of this dataset. I had always considered it to be 'just tabular' but I realize that there is a hierarchy and geographic data (and a geographic hierarchy) which I hadn't really considered before. As I type this, we could layer in time and sets when considering multiple elections." Data abstraction typologies can help data workers discover underlying latent abstractions, like hierarchies, or how visualizing their data with additional data abstractions may augment understanding, like adding images to patient records.

C24. Data abstraction typologies help data workers communicate at a sufficient level of detail to design a visualization system. We observed this directly with I6–I12. A survey participant also noted that the mental exercise of the survey "prodded me into thinking about my annotations as more of a central player in the overall visualization as opposed to a secondary thought or supporting contextual element." Discussing abstraction typologies helps create a common data design language and reinforces the value that both sides (the data worker and visualization designer) bring to the data problem.

4.5 Themes

Together, these codes form four overarching themes, including the prevalence of latent data abstractions, the interventionist impacts that pursuing latent abstractions can have, why many data workers may express hesitancy to pursue latent abstractions, and the benefits that transparency about data typologies can have for the latent abstraction discovery process. Here, we enumerate the evidence that supports each theme.

T1: Latent data abstractions are very common. At least initially, raw data formats are not designed in such a way as to anticipate all abstractions that may be needed or useful, yet even though these abstractions may not be fully actualized in a computer, data workers are often aware of meaningful, useful abstractions that they can communicate about without specific prompts **(C7)**. Some of these abstractions, particularly networks **(C11)** and groups **(C12)**, are very intuitive to many data workers.

This theme validates a known [133, 10] phenomenon that data rarely has a "correct" abstraction, even where predominant file formats exist; we observed that discrepancies between raw file formats and the way that a data worker thinks about their data are common **(C1)**. Instead, data abstractions have a complex and evolving form **(C14)** that must be explicitly designed.

The designed nature of data abstractions makes it important to note that neither data workers nor theorists possess comprehensive knowledge of all possible latent abstractions, and open-minded communication is necessary for meaningful, useful abstractions to be discovered. This is true for both parties: theorists are often aware of abstractions that data workers might not consider to "count" as data **(C8)**. Similarly, data workers may be aware of abstractions that theorists do not consider to "count" as data **(C17)** **(C18)**. Data workers and theorists may also think about the details of the same abstraction differently **(C13)**. Introducing a typology of data abstractions can expose abstractions that neither party has considered, in that a typology can contain new abstractions that data workers may not be aware of, or they may lack new abstractions that theorists have not considered **(C23)**.

T2**: The visualization community identifies data abstractions for its own transferability needs, but the process of identifying an abstraction is an intervention with far-reaching effects.** Collaborations with data workers beyond the visualization research community stand to benefit—and can be harmed—by the way that both parties introduce, articulate, and explore data abstractions.

Our data validates that visualization researchers, as theorists, are not operating in a vacuum; some abstractions that are common in the research community are intuitive to many data workers **(C11)** **(C12)**.

However, although these commonalities may be good news for the validity of the work that visualization researchers perform, there are also areas in which the culture of visualization research clashes with data workers at large: there is a often a disconnect between what theorists consider to be data and what data workers consider to be data **(C8)** **(C17)** **(C18)**. Disconnects also occur between the language that theorists use to describe data, and the language that data workers use **(C16)**. These differences in culture risk miscommunication at best, but also may put a project at risk of devolving into a bad collaboration, where either the goals of the theorist or the goals of the data worker become subordinate.

Consequently, for better or worse, introducing a theoretical perspective is almost always an intervention, and the effects of such interventions can be profound. Because the design of data abstractions is so inextricably linked to the other concerns of data discovery, capture, curation, and creation **(C21)**, changes to the design of a dataset can result in changes to all of its other aspects. Similarly, influencing a data worker's mental model of their data can have far-reaching practical effects, including disruptions in workflows and changes to the file formats **(C19)** and software **(C20)** that data workers use.

Data workers are often cognizant of the impacts that changes to the design of their abstraction will have **(C9)**, even if they may not be able to fully articulate these impacts in detail **(C10)**.

This is why we predict that T3**: data workers are less willing to pursue**

latent data abstractions when the design of an existing abstraction is already fundamental to their workflow. When there exists a direct mapping between familiar software and the format of the raw data, efforts to introduce a new abstraction will likely be met with resistance.

The costs of a changed data abstraction design can include a need to learn new file formats **(C19)** and new software **(C20)** that may come with the need to learn new software skills such as programming. The tight coupling between data abstractions, workflows, and software can be seen in the bespoke wrangling software needs that arise from the combinatoric expansion of diverse abstractions, diverse workflows, and diverse dataset scales **(C6)** **(C2)**. However, the added cost is reduced when software practices have not yet been established and investments in learning new skills have not been made. This cost can also be mitigated when theorists are willing and able to provide expert help **(C4)**, such as wrangling the data to its needed forms.

Similarly, the costs of pursuing latent data abstractions can propagate to other data concerns **(C21)**, such as the need to collect additional data **(C5)**. The fears that data workers often feel **(C10)** and voice **(C9)** are suggestive that data abstraction changes can spill over into task abstraction changes that may begin to depart from data workers' actual needs. This potential cost can actually be an opportunity if care is taken to solicit critique whenever theoretical perspectives are introduced, as such introductions often encourage data workers to provide detailed information about their mental models that they might not otherwise articulate.

Theorists need not wait for such impositions, however, to solicit this kind of targeted feedback. T4**: Like access to real data, introducing a data abstraction typology helps to focus reflection and communication about data abstractions at a level of detail that includes actionable information.**

Our data **(C22)** validates the known pitfall [133] in which the lack of access to real data can doom a design study collaboration, because visualization researchers are less likely to have enough actionable information to articulate an accurate data abstraction. It also validates that a culture of data review [156], that is careful to emphasize good communication and transparency about the data abstraction, can compensate for a lack of access to real data because the detailed abstraction is a joint objective that all parties have a stake in.

When theorists take the time to be transparent about their agenda, including the typology that they are attempting to fit a worker's data into, revealing the typology can have similar benefits in that it helps a data worker understand what a theorist is looking for **(C24)**. Introducing typologies can expose data workers to latent abstractions that they may not have considered **(C23)**, and provides an opportunity to provide detailed feedback that might otherwise be left unspoken **(C17)** **(C18)** **(C13)**. For example, introducing a typology that is a poor fit in how it subdivides data abstraction categories can serve as an aid to communication, in that it can highlight the detailed ways that a worker considers their data to fit or partially fit more than one abstraction category **(C14)**.

Not all shortcomings of a typology are equally beneficial, however. Data abstraction categories that are too general **(C15)** or rely too heavily upon jargon **(C16)** may have limited utility. These limits are highly contextual, for example, a typology that differentiates between partitions of an abstract mathematical space and regions of a physical three-dimensional space might be useful for a data worker with a rich mathematical background to reflect upon; however, for a worker with less mathematical training, the amount of unfamiliar jargon introduced could inhibit detailed feedback.

When introducing a data abstraction typology as an explicit design activity [95], care should be taken to choose a typology with an appropriate level of granularity and enough accessible concepts to encourage feedback and critique.

4.6 Discussion

The codes and themes that we present describe phenomena that are suggestive of guidelines for theorizing about data abstractions. Additionally, it has implications for reporting data abstractions in many kinds of visualization research. We also reflect on our experiences and their implications for the design of data abstraction typologies, and lessons learned from our innovations in survey design and deployment.

4.6.1 Guidelines for Pursuing (Latent) Data Abstractions

Reflecting on the presence of latent data abstractions **(T1)**, the interventionist nature of defining data abstractions **(T2)** and in some cases the resistance to it **(T3)**, and the focusing power of typologies **(T4)**, along with our coded findings in section 4.4, we proffer the following guidelines:

Data owners and abstraction theorists should collaboratively probe raw data. A typical design workflow may have data owners *describe* their data synchronously and then *give* one or more data files to the abstraction theorists for later review. There are several surfaces of loss in this approach, in which latent information remains latent. Data owners may forget to review elements of their data. Abstraction theorists may make assumptions given the data file that are only revisited much later, if at all. Instead, we recommend that initial meetings with data owners involve the presentation and collaborative probing of at least one raw dataset.

Abstraction theorists should introduce the typology and process that they follow. Just as theorists can feel lost without exposure to the raw data, data workers can feel lost when theorists attempt to fit a worker's project into an opaque typology or framework. For example, if a worker does not understand, at least at a basic level, that a theorist is attempting to identify relevant data abstractions before considering visual encodings, workers are forced to second-guess the theorist's needs. In such a situation, discussing their data in terms of potential visualization designs may appear to be *helpful*. As theorists request that workers provide at least one raw dataset, theorists should also reciprocate by preparing and presenting sufficient background about what they are hoping to learn or observe.

Create artifacts that document and convey abstraction details and demonstrate possible permutations. We discovered that even in discussion among the authors, people who had a close working relationship and were operating from the same typology, there were times when we believed we were discussing the same abstraction of the data, only to discover we had completely different assumptions once drawings or classifications were made explicit. Explicitly stating ideas serves as not only a communication aid, but also as a method to explore the creative space of possible abstractions and as documentation for resulting abstractions. Furthermore, writing or drawing such low-level details can be an effective strategy to ground a derailed conversation and refocus it back on the design of the data.

Challenges are an effective means of probing. They require an artifact to be challenged. Throughout our interactions with data workers, we observed that suggesting a concrete abstraction, particularly one that was unlike how the data worker usually conceptualized their data, elicited rich feedback about their data and their thinking on it. Responses beginning with phrases like "That wouldn't work because..." or "That makes no sense" were precursors to valuable reflections on their data. Setting up such a response requires some form of artifact, verbal, pictorial, textual, or otherwise to be challenged. We recommend such situations be approached sincerely as an honest, creative exercise towards considering other forms.

Typologies can serve as a guide to elicit latent elements of the data abstraction from data workers. A given typology may not fit all elements of a particular problem and dataset. However, it provides a corpus of possible abstractions with which to consider the data. These possibilities can serve as a jumping point to discuss and challenge possible abstractions of the data. Through our survey and interviews, we observed that discussions of fitting the data to various forms evoked more detail about the data itself as well as provided structure to exploring possible alternative abstractions.

Document and share the provenance of datasets. It is appropriate that a visualization and analysis solution operates on a brief period of a dataset's lifecycle and often only a subset of all possible data available. However, it is beneficial to document the latent elements of the data beyond that directly used by that solution. The source of the data, the transformations it has gone through, and related data all provide context. This context can be used to better understand how the data worker, who is more familiar with all of these elements, conceptualizes the dataset.

Assess opportunities inherent in derailments. The space of (possibly latent) data abstractions is vast in comparison to the minimal data abstraction represented in a visualization project. In following these guidelines, it can be easy for both theorists and workers to feel that the discussion has become derailed: workers may begin to discuss other data concerns such as data discovery, capture, curation, or creation. Workers may also discuss specific software or even prematurely begin to volunteer visualization encodings and techniques. Similarly, theorists may appear to be exploring esoteric concepts that do not have a clear application to a worker's project, and their explorations may threaten to add unnecessary labor to a worker's workload.

These derailments can be an opportunity to gain insight: First, discussing the design of a dataset has a tendency to prompt communication of important low-level information—even if seemingly unrelated—that workers would not otherwise bring up. Second, workers may actually be speaking on topic, but using seemingly irrelevant language about formats, software, or visualization as proxies that can be revealing about domain conventions or language, as well as revealing a need for the theorist to be more transparent about what they are looking for. Third, seemingly irrelevant topics may be indicative of a high-level mismatch of objectives, differences in perspective, or other miscommunications that could otherwise go unnoticed.

Actively seeking critique from data workers can help to identify a theorist's own derailments. Once derailments are identified, ascertaining the extent to which any of these three opportunities exist can help guide a theorist as to whether, when, and how to attempt to recenter the conversation.

Document objectives and revisit them regularly. Collaborators often have different high-level expectations, ideas, agendas, and sub-goals/tasks. This is complicated by the potential for a latent abstraction—even considering one hypothetically—to change collaborator perspectives and goals in ways that may not be communicated immediately. We recommend documenting the objectives of the project, and revisiting those objectives, especially when derailments are indicative of high-level mismatches.

Schedule interventions to revisit data abstractions. The above guidelines discuss how to make the latent *apparent*, but require the latent exist in the minds of people or the artifacts (e.g., the raw data) available. However, over the course of the project, all people involved may discover new facets of the data or incorrect assumptions previously made. Sometimes these discoveries lead to immediate intervention, but sometimes they expand the latent space. We recommend scheduling time to revisit, challenge, and refine data abstractions, given possible discoveries that are latent.

4.6.2 Implications for Reporting Data Abstractions

Our data suggests that providing the expert help that many data workers need can make visualization researchers more effective collaborators. Until recently, as we discuss in subsection 4.2.3, performing, documenting, and reporting on this kind of work may have been difficult to accomplish by itself, even though there is a great need for published guidance and experience to inform many different kinds of visualization research.

We expect performing and reporting on detailed, applied data wrangling work better equips visualization experts to collaborate effectively. Recent acknowledgements of “Data Transformation,” [105] “Data Abstraction,” and “Data Structure” [86] as potential standalone contribution areas may aid in these efforts. We also suggest that such reports may be able to help ground technique- and systems-focused research in more evidence-based user needs.

4.6.3 Implications for Designing Abstraction Typologies

Our experience in attempting to apply the same data abstraction typology to a diverse array of data workers and datasets revealed wide variability in the extent to which typologies are likely to fit a particular context—both the diversity of datasets and the diversity of data worker expertise and perspectives can risk a poor fit.

Our data shows that this is not necessarily problematic. It demonstrates how typologies can be useful in pursuing latent data abstractions despite—and, in some circumstances, *because* of—their limitations. In the spirit of the observation that "all models are wrong but some are useful" [17], shortcomings of a typology can create opportunities to aid in detailed communication and reflection that might be less likely if the typology were a perfect fit.

This also suggests that typologies may not scale well for purposes beyond the pursuit of latent data abstractions: typologies must generalize in order to be tractable and support comparison, however, generalizations fundamentally censor diverse, individual voices and risk stifling important exceptions and innovative thinking. Our corpus of survey responses demonstrates a way that a conversation about the nature of data abstractions can be conducted at scale, in a way that balances the need for generalizability, while giving priority to individual viewpoints and grounding discussion in the context of real-world applications. In the way that our survey explicitly sought critique on the typology that we presented, it allowed for enough organization to visualize, compare, and contrast hundreds of viewpoints, while giving wider freedom for participants to engage directly with the theoretical questions implicit in the design of the survey.

4.6.4 Reflections on Survey Innovations and Deployment

Unlike typical surveys that primarily collect quantitative information for well-defined questions, our main objective in deploying the survey was to probe for blind spots in our own understanding of what data abstractions exist, and how data workers think about them.

Consequently, we sought to create a survey that was as open-ended as possible. Closed questions are therefore least ideal, as they provide zero opportunities for a participant to signal to researchers that the researchers may have missed something—researchers have to anticipate every possible response [122].

Open-ended, free response questions at least make it possible for participants to submit critique, but because they're expensive to code and analyze, and because they introduce more survey fatigue, they often take the form of a single comment field at the end that are only used as an "outlet" for participants, rather than a prioritized source of data [48].

The extent to which participants freely made use of the ability to skip survey sections suggests that this approach has several benefits. Replacing a whole section of a survey with a single free response field appears to help mitigate survey fatigue. The free response field is at least as open-ended as regular free response questions, and consequently incurs no additional analysis cost. The act of stepping outside the normal flow of the survey appears to have encouraged participants to think about the design of the survey itself, and in some cases, engage at a theoretical level that more closely resembles a forum than a survey.

In contrast, our interactive glossary did not appear to have garnered as much attention or use—this may have been due to its placement outside the flow of the survey, and/or its position in the corner of the screen.

The survey innovations created opportunities to improve our understanding of what data abstractions exist, what terminology is actually used by diverse data workers, to refine the evolving themes, and we expect it will inform future iterations of the survey.

4.7 Limitations and Future Work

Here we document the limitations of the survey that we present, our intent to deploy it to a broader audience, and suggest future uses for the dataset that we have released.

4.7.1 Survey Design and Evaluation

The archive of survey responses that we present is not without typical technical difficulties. One major drawback of its design was that the length of the survey varied, depending on the difference between “rarely” thinking about a dataset as a certain type and “never.” This resulted in some participants filling out lengthier surveys, who began to show signs of fatigue. Additionally, a question in several of the Details sections had a bug that failed to capture the data completely. Finally, as participants almost always took the survey on their own devices, connectivity and browser incompatibility issues, especially for specific iOS devices, occasionally arose. These challenges, together with a small number of responses in which participants appeared to abuse the ability to skip sections, etc., resulted in a set of questionable responses that we have flagged.

Rather than suppress these errata, they are included in the archive of the data, and documented in context in the visualized summary of each question. The set of questionable responses can also be interactively filtered out.

As the survey design itself is not our primary contribution, we have only evaluated the extent to which our innovations were effective in achieving our qualitative aims. We can not speak to whether they are effective ways to solicit critique from participants in general, nor engaging enough to encourage theoretical reflection at the levels that we observed.

4.7.2 Further Survey Deployment

The feedback that we collected may also have been influenced by the groups where we deployed the survey and wrote memos about our observations. The populations we engaged with during this study all had a high interest in computing: domain scientists who come to hacking-oriented meetups and attendees at computing conferences. Although the Supercomputing conference has thousands of attendees who are there for reasons other than the technical program, in some interactions, we had difficulty convincing those people that their data counts as “data.” Thus, our data and subsequent findings are lacking representation among people who do not identify with data.

Effectively engaging people with less overt interest [114], that may not share the goals represented by our "data worker" persona, is an ongoing effort that we hope to pursue in future work. Subsequent survey deployment and memo writing will target more diverse data perspectives and skill sets, by networking with people from non-Computer Science backgrounds. For example, Meetup attendees have already referenced ongoing discussions about data abstractions in a paleontology community. They are considering how to best match and connect competing ontologies from different sources. Similarly, we have been connected with a group of vehicle mechanics that are adapting their tables of diagnostic metrics to changes introduced by increasing numbers of electric vehicles. Other potential domains include linguistics, sociology, bioinformatics, construction equipment and manufacturing, and athletics. We intend to advertise and deploy our survey to more diverse groups of data workers, through academic and professional conferences, at relevant community Meetup events, and through word of mouth.

4.7.3 Data Reuse

We have released the public portions of the survey data in a visual, searchable format as a standalone research contribution, so that individual voices can be heard and reviewed by researchers studying similar phenomena, beyond our research aims. Such aims might include creating terminology maps across domains, using evidence in our survey responses to motivate and justify the design of general-purpose visualization and data wrangling tools, and other analyses.

4.8 Conclusion

Our grounded theory investigation into the malleability of data abstractions has resulted in themes that describe data abstractions and their implications for visualization design, guidelines for the development of data abstractions, the design and deployment of an open survey, and a corpus of survey responses that represent a discussion about the nature of data abstractions at scale. This work has implications for how data abstractions are reported, how typologies are designed and discussed, and may inform future efforts to seek critique through the deployment of surveys. Ultimately, this work sheds light on why thinking and communicating about data abstractions can be difficult, and shows how to best take advantage of opportunities inherent in that process, as well as mitigate its risks.

Family and Friends Summary: In the tree visualization from section 3.8, we chose to omit some lines and visually present the data as a tree, instead of a graph, because our users knew the underlying data was a graph but found the tree to be a more streamlined view of the data. This situation became part of larger discussions about how easy it is to change *data abstractions*. A data abstraction is way to generalize a specific dataset into a more abstract form. For example, a list of contacts in a phone can be generalized as a tabular data abstraction, i.e. a table of names and phone numbers. Visualization designers know how to create visualizations from tables, so they can tweak an existing solution for this specific case. Alternatively, the list of contacts could be cast in a network or graph data abstraction, where each node is a person and their phone number, and edges between people represent relationships.

We were curious if data experts would find it useful to think about their data in a different abstraction, like with the phone contacts. To investigate, we interviewed and surveyed data workers and asked them to consider different data abstractions for their dataset. We found that the participants often could think of their data in an alternative data abstraction (a *latent data abstraction*), but were not always aware that this alternative existed. Our data abstraction options (i.e. our *data abstraction typology*) was a helpful starting point for these discussions, and occasionally provoked strong reactions. The data workers found these discussions and latent abstractions to be useful, but admitted they were less willing to pursue the alternative abstraction since it did not fit into their existing workflow. This work has implications for how data visualization designers discuss data and explore data abstractions with their users.

CHAPTER 5

Data Abstraction Elephants: The Initial Diversity of Data Idioms and Mental Models

The elusive nature of data abstractions and their implications for a dataset means that for visualization designers, the choice of data abstraction can significantly impact how our users understand a data visualization. A visualization designer may choose one data abstraction as the basis of the visualization, but the users may think of the data in a different data abstraction. For example, the user may give the designer a tabular dataset. However, the user may understand connections between the data that are not explicitly in the file provided to the designer. These hidden connections may be best represented for this user as a graph or network.

This scenario reminded us of two stories involving elephants. The first is the parable [139] of the five people in the dark who encounter an elephant for the first time–they all touch a different part of the elephant and come to a different conclusion. The person who feels the tusk says the elephant is hard and smooth like a spear, and the one who touches the leg says the elephant resembles a tree trunk. This story mirrors how each person may have a different mental model of the data, shaped by their prior knowledge, expectations, and inferences.

The second is Von Neumann's purported observation, "With four parameters I can fit an elephant, and with five I can make him wiggle his trunk;" in other words, the data can be made to fit what we want to see. These stories implicate possible pitfalls that can arise when applying different data abstractions.

In this chapter, we seek to understand the space of data abstractions used in mental models and how well people communicate their personal mental models through sketching and discussion. The choice of data abstraction has significant impact on the visualization design [103], yet it is unclear how universal this choice may be when not influenced by the initial data representation.

We conducted a study of how people create their mental models from a dataset.[1] We presented each participant with one of three datasets in paragraph form so as to avoid any visual cues that may influence the data abstraction and mental model. We observe a variety of mental models, data abstractions, and depictions from the same dataset (see Figure 5.6). This diversity was influenced by several factors including examples the participant had recently seen, common approaches to the data, imagined purpose-seeking tasks, and their definition of what "data" is. We also observed participants re-configuring their mental model and sketch when considering describing the data to another person, and they used a variety of terms to describe the data and sketches.

Our results have implications for visualization design, especially in the discovery and data collection phases when eliciting descriptions of data. We propose leveraging the data design process to further probe user needs and suggesting alternate perspectives to users to elicit possible abstractions.

This work was previously presented at CHI 2023 and the published work is available [155][2]. The content has been modified slightly to fit the flow of this dissertation. This work was done by Alex Bigelow, Katherine Isaacs, and myself and was supported by NSF-1844573.

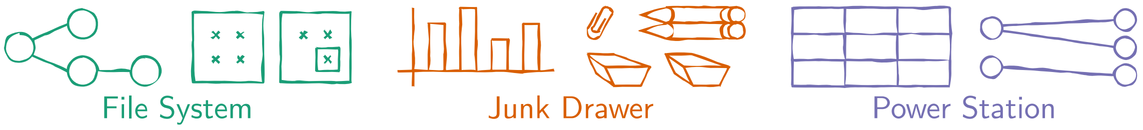


Figure 5.1: We created three datasets that permit a variety of different data abstractions. The file system dataset sketches include hierarchies and nested sets. The junk drawer dataset sketches include bar charts and drawings of the physical objects. The power station dataset sketches include tables and node-link graphs. We found a variety of data abstractions across each dataset.

[1]Our session notes and participant sketches are at `https://osf.io/kvnb9/`

[2]`https://doi.org/10.1145/3544548.3580669`

5.1 Introduction

A viewer of a data visualization brings their wisdom, experiences, biases, and interests to their viewing. This internal knowledge and their understanding of the data visualization comprise their mental model of the visualization [58]. A mental model is a personal understanding of a topic that may consist of representations of objects, background knowledge about the topic, and connections to related topics. In the field of data visualization, research has been done on mental models arising from dashboards of political data [131], trees and hierarchies [163], social networks [137], and scientific visualizations [142]. But what about the mental model that exists *before* the visualization is made, when the visualization designer and the domain expert are discussing the dataset? As visualization designers, what steps should we take to elicit and understand our viewer's mental model and how should we design following that mental model to maximize understanding and utility?

For visualization designers, our usual starting point with a new dataset is to connect it with an existing data abstraction, like a table or a network. A data abstraction is a mapping of domain-specific data to an abstract data type [104]. By selecting a data abstraction that has been repeatedly used and refined, we narrow the scope of possible visualizations to create and increase the likelihood of success by building on others' prior work in visualization. A data abstraction provides an intermediary for the designer and viewer, providing a structure for the viewer's intangible mental model and guiding the designer toward visualization design choices that will resonate with the viewer. However, often there is more than one data abstraction that may work for a given dataset. The same dataset may be initially matched with a hierarchical data abstraction, but a set data abstraction could work instead. While sometimes there is a *better* data abstraction choice for a dataset, more likely there is simply an alternative data abstraction that provides different insights.

The variety of abstractions and their implications for visualization reminded us of two stories involving elephants: the parable of the five people in the dark, reaching different conclusions about the nature of the elephant; and John von Neumann's overfitting of data to an elephant, whether or not the data truly represents one.

The parable, which has appeared in Hinduism, Jainism, Sufism, and Buddhism [144], describes five people who are unable to see encountering an elephant for the first time. Each person touches a different part of the elephant and comes to a different conclusion. For example, the person who feels the tusk says the elephant is hard and smooth like a spear, and the one who feels its side thinks it is like a wall.

Von Neumann's purported observation was, "With four parameters I can fit an elephant, and with five I can make him wiggle his trunk;" in other words, the data can be made to fit what we want to see.

These stories reflect possible pitfalls when applying data abstractions to a dataset. The first reminds us that each person may have a different mental model of the data, shaped by their expectations, prior knowledge, and inferences; this mental model impacts their preferred choice of data abstraction. The second reminds us that an individual, perhaps a data visualization designer, can force-fit a data abstraction where it may be unhelpful or misleading. With these pitfalls in mind, we set out to better understand the breadth and form of data abstractions arising from people's mental models and how they communicate their mental models before they are influenced by abstractions or representations chosen by other parties. We seek to build a foundational understanding to drive more concrete guidelines and methodologies for eliciting mental models and exploring data abstractions during the design phase of visualization design [133].

As visualization researchers, we recognize that our users may have mental models of the dataset that could prove to be valuable resources to leverage during the design process. Often in the case of design studies, we are creating a visualization tool where none has previously existed. The only mental model the user has is one of the data space and their interactions with the dataset. The user's mental model may include aspects of an insufficient visualization that serves some but not all of the user's needs. In situations like these, the data and tasks are often still fluid and need to be stabilized.

This instability in the data and tasks during the initial stages of the design process can be beneficial, as it provides more options to explore and does not impose a bias on the design. As we show in this paper, the creativity and lateral thinking shown during our interviews about mental models of datasets suggest that domain experts can offer creativity coupled with domain knowledge that could lead to more productive brainstorming and collaborating early in the visualization design process. However, the best practices for eliciting mental models of data and incorporating their related data abstractions in visualization design are unclear.

As a visualization community, we would like to develop more concrete guidelines and methodologies for eliciting mental models to help steer our data abstraction choices. However, we need to understand fundamentally how internal representations of data are translated to external representations and how difficult pinning down that mapping can be. To begin, we consult existing literature on mental models and their elicitation in areas like education, natural resource management, artificial intelligence, cognitive science, and psychology.

Mental models are notoriously difficult cognitive phenomena to elicit [31]. Klein and Hoffman describe the multitude of reasons why we should not study mental models, yet argue that *because* of their slippery, elusive behavior, we should continue to strive to find best practices for eliciting, describing, and analyzing mental models [81]. We continue the conversation by asking these questions to further understand how understanding mental models can help with visualization design: How do we avoid choosing a non-fruitful data abstraction during visualization design? How many abstractions should we include if multiple abstractions provide insight into the data [13]? There is an inclination toward selecting a single "good" abstraction, but by doing so, how much do we compress the space of reasonable abstractions? Is there a breadth in how people think about these data abstractions in their existing mental model of a dataset? How big is this breadth? We do not attempt to answer all of these questions but provide this paper as a starting point for the community to investigate mental models at the start of the design study process, before the existence of a visualization, to strategically explore suitable data abstractions.

Specifically, we begin by asking the following research questions:

- What factors influence people's initial mental models of data?
- What encodings and visualizations do people commonly use to communicate their mental model?
- How do people describe how they think about the data? How do people describe their sketches?
- How difficult is it for people to sketch and/or describe their mental model? How difficult is it for us to understand?

With the answers to these questions, we can have a better understanding of how users attempt to convey their mental models of datasets, which allows us to incorporate aspects of their mental model in our choice of data abstraction and visualization design. While no one use case is a perfect representative of a "typical" design study, we conduct this experiment using small, incomplete datasets in paragraph form to represent design studies where the data are evolving and both the designer's and the user's mental models of the data are shifting throughout the design process. Further research is needed into techniques for improving the elicitation process in a visualization design context, but our study shows how semi-structured interviews and eliciting representations in the form of sketches can be effective means of clarifying how a person thinks about a dataset. These results have implications for how designers approach the initial stages of the design study methodology, and how effectively and efficiently they can execute the design study.

Recognizing the open-ended nature of these research questions, we conducted a study into the mental models and data abstractions people create from a dataset. Rather than presenting participants with tabular data, which has been shown to influence design choices [10], we presented participants with one of three datasets in paragraph form. We observed a wide variety of mental models, data abstractions, and depictions from the same dataset, as well as how these concepts are influenced by communication and purpose-seeking. We present our collection of core concepts and their implications for visualization design.

In summary, our contributions are:

1. A set of themes, supported by codes, that describe the diversity of initial mental models and data abstractions, their depictions and influences leading to them, and how they are communicated (section 5.4),
2. Implications of these themes and codes for visualization and data design (subsection 5.5.3), and
3. An open database of the sketches and transcripts resulting from the study. [3]

We discuss background in mental models, data abstractions, and sketching (section 5.2). Next, we detail our study methodology, the motivations behind our three synthetic datasets, and our analysis process (section 5.3). We explain how our interviews and sketches support our codes, which in turn motivate our themes (section 5.4). We discuss our research questions (subsection 5.5.1), and the limitations of our study (subsection 5.5.2), and we provide implications for the visualization community (subsection 5.5.3).

5.2 Background and Related Work

We discuss related work in mental models, data abstractions, and sketching in visualization.

5.2.1 Mental Models

We draw on the abundance of research on mental models in areas like cognitive psychology [69], design [62], and HCI (e.g., [55, 137]), as well as their applications in visualization (e.g., [92, 14]), natural resource management [71], computer science education [52, 128], and engineering [58]. A *mental model* is an individual's

[3] `https://osf.io/kvnb9/`

understanding of a subject or concept that consists of their prior knowledge, understanding of the presented material, and integration of the knowledge with their worldview. Mental models are more abstract than perceptual images. They contain less detail because our brains omit details we deem irrelevant, yet contain more information than a visual image because they include our prior knowledge [129].

Research on mental models often examines how well people learn "something in the world," frequently an interactive or dynamic system [81]. Klein and Hoffman explain that the mental model is shaped by the rules, laws, and principles that govern this "something" as we observe and learn how this something exists in the world [81]. Jonassen and Henning state mental models are "representations of objects or events in systems and the structural relationships between those objects and events?" [70]. We compare our findings to those of other mental models in subsection 5.4.7.

Studying mental models is challenging because there is limited accuracy, they are unique to each individual, they are incomplete representations of reality, they are inconsistent and context-dependent, and are highly dynamic models [71]. Klein and Hoffman outline these issues with mental models, and why researching mental models can be controversial but worthwhile [81]. They argue that it is imperative to understand how mental models are formed and how mental models may be modified to increase the depth of understanding, with applications in education and group discussions.

Given mental models are internal phenomena, any method to elicit a mental model can only give us a representation of the mental model. Sketching, interviews, and arranging topic cards are common due to their flexibility. Harper and Dorton created a more specific elicitation method for mental models that uses a detailed notational framework to visualize the mental models [55]. A more indirect approach is observation, such as listening to participants think aloud about their strategy in the word-guessing game Passcode, as they work with an AI to understand how the AI gives and receives clues about the word [49]. Regardless of the method, these knowledge-elicitation methods have been repeatedly tested by cognitive scientists and the strengths and weaknesses of using the methods on mental models have been discussed at length [30, 31, 71].

A popular strategy for studying mental models is to use direct elicitation. Direct elicitation requires the interviewees to represent their understanding of a given topic externally, e.g., by drawing a diagram of their mental model or by arranging a set of cards of existing concepts [71]. Interviews are also viable ways to elicit mental models. Milgram and Jodelet asked Parisians to draw a map of Paris and speak about all of the elements of the city that came to mind. From the activity and follow-up interview, they found that participants' sketches of "their" city were a combination of major city landmarks and personal touches, such as a butcher including the meat stockyards or an architect adding an avenue to connect prominent structures [99]. Like with all representations, these representations of mental models are influenced by the skill of the interviewer and the ability of the interviewee to verbalize their understanding.

5.2.2 Mental Models and Data Visualization

In a collaborative group setting, people share ideas and socially negotiate a community mental model that draws on collective experiences, knowledge, and wisdom from the individuals in the group [70]. This setting occurs in the use of data visualizations, such as when stakeholders are analyzing a visualization. This collaboration also occurs in the early stages of the design methodology, when domain experts and designers are negotiating the data and tasks they wish to support. Liu and Stasko argue for the inclusion of mental model research in visualization, saying that visualization can be viewed as a tool to support the formation of mental models about data and information [91]. They developed a visualization-centric definition of a mental model, stating a mental model is a "functional analog representation to an external interactive visualization system" and listing characteristics of that internal representation. They use this definition to explain how internal representations affect how people interact with external representations and vice versa. To put this theory into practice, Mayr et al. present measures and evaluation procedures to assess mental models in other domains and discuss their applicability to information visualization [92].

Visualizations are effective ways of modifying mental models to improve understanding. The addition of an effective visualization when learning new concepts can be critical to developing a viable mental model of a new subject or system, such as computer architecture [160].

5.2.3 Data Abstractions

A *data abstraction* is a mapping of domain-specific data to an abstract data type [104], e.g., power station supply lines can be mapped to a network, providing a more generalizable form to the data. Abstraction should happen early in the design process, during the discovery stage, and should be frequently re-examined by the domain experts to ensure correctness and cohesion with their mental model of the problem [133]. The mental model of the user might not neatly correspond to one particular data abstraction, but the discussion around the data abstraction can serve as a way for the user to make their abstract mental model more concrete to help the visualization designer. The visualization designer may need to change the abstraction based on their understanding of how the user interacts with the data and the tasks they are trying to accomplish. Exploring alternative abstractions and their usefulness is much simpler at the beginning of the design process before significant time and resources have been invested. Often there is not a single correct abstraction; instead, abstractions must be designed [98, 102] to best suit the user's needs.

Many authors have identified that difficulties exist in communicating effectively about data abstractions [119, 133, 149]. Trees and graphs can be especially hazardous abstractions to work with, in terms of their potential for miscommunication [108], especially when dealing with edge cases or when people use mathematically imprecise language to discuss graphs [45]. Bigelow et al. found that introducing a *data abstraction typology*, a model that describes the space of possible data abstractions and/or data wrangling operations, can spark discussion and elicit more specific communication about the dataset and abstraction, even when the typology is imperfect [13]. Similar to Bigelow et al. [13], we conduct a study of data abstractions; however, we seek to answer a different set of questions. Bigelow et al. focused on the utility of considering a change in the dataset type of an existing data abstraction. We seek to understand how multi-abstraction datasets can be interpreted and represented.

Tension naturally arises when trying to work with data: tension between the internal data abstraction and the external data abstraction, tension between the imagined visualization and the constraints of the system [13], and tension between the provided data and the desired data. Tension between users and visualization designers may also arise. Even visualization designers and developers may have difficulty communicating about data mappings, anticipating changes to the data, and elucidating technical challenges [149].

5.2.4 Sketching

Sketching is used in different ways in visualization, often for prototype demonstrations by designers, but also in understanding how people create visualizations for their own personal use. Data sketching is a simple way to show personal mental models, such as students' concepts of time [44] or homeowners' concepts of their home wireless network [118]. Understanding the language of diagrams and how we visualize our thoughts [143] enables us to successfully collaborate and share visualizations. Communication and gestures help augment what is on the page [26].

Walny et al. used data sketching to examine *external representations* people created from a novel dataset [151]. They examined the diversity of data representations and the relationship between sketches and people's understanding of that data. Participants were given a table of ratings of human behaviors in social settings as a dataset.

While our study shares similarities with Walny et al. [151], there are significant distinctions between the two. For methodology, we presented our dataset in paragraph form, rather than in a table, to minimize influencing the data abstraction with a prior data abstraction. Text is not a typical format of the data, but the paragraphs were rather list-like (see subsection 5.5.2). Our study had a pre- and post-sketching discussion, rather than writing a free-response answer to a question. For research questions, Walny et al. examined the range of visualizations that were created, placing the participants' sketches on a numeracy to abstractness continuum. We use this numeracy to abstractness continuum to code our results, see subsection 5.4.6. However, rather than the encodings and contents of the sketches themselves, we are more interested in what sketched representations can reveal about the data abstractions that participants assume or construct in their minds. As we were interested in mental models and views about data, our semi-structured interviews allowed us to delve into these discussions.

5.3 Study Methodology

To elicit data related to data abstractions, we conducted interview sessions where participants were asked to sketch a small dataset and then discuss their sketch and mental model through a semi-structured interview. We used three datasets designed for the potential to elicit different data abstractions, with each participant being shown one. We piloted the study with five participants, after which we iterated on the designed datasets and the interview questions.

Three authors participated in coding interview transcripts and sketch photographs and met regularly to develop codes further. We continued collecting data until we reached saturation regarding our research questions. We describe the details of this study below. An overview of our procedure is shown in Figure 5.2.

5.3.1 Participants

We recruited 28 participants, listed in Table 5.1 by occupation and dataset prompt. We sent out recruitment requests to five organizations, of which we recruited participants from a university's computer science (CS) Discord server and undergrad CS mailing list as well as posting fliers and word-of-mouth in the local YMCA community. Of the participants, 20 had computer science-related work (16 were CS

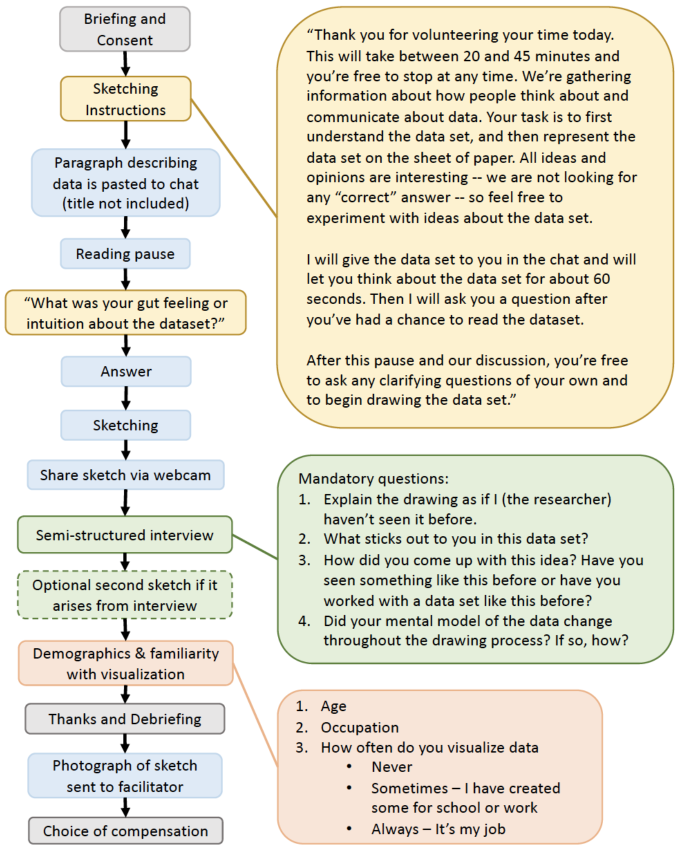

Briefing and Consent
Sketching Instructions
Paragraph describing data is pasted to chat (title not included)
Reading pause
"What was your gut feeling or intuition about the dataset?"
Answer
Sketching
Share sketch via webcam
Semi-structured interview
Optional second sketch if it arises from interview
Demographics & familiarity with visualization
Thanks and Debriefing
Photograph of sketch sent to facilitator
Choice of compensation
"Thank you for volunteering your time today. This will take between 20 and 45 minutes and you're free to stop at any time. We're gathering information about how people think about and communicate about data. Your task is to first understand the data set, and then represent the data set on the sheet of paper. All ideas and opinions are interesting -- we are not looking for any "correct" answer -- so feel free to experiment with ideas about the data set.
I will give the data set to you in the chat and will let you think about the data set for about 60 seconds. Then I will ask you a question after you've had a chance to read the dataset.
After this pause and our discussion, you're free to ask any clarifying questions of your own and to begin drawing the data set."
Mandatory questions:
1. Explain the drawing as if I (the researcher) haven't seen it before.
2. What sticks out to you in this data set?
3. How did you come up with this idea? Have you seen something like this before or have you worked with a data set like this before?
4. Did your mental model of the data change throughout the drawing process? If so, how?
1. Age
2. Occupation
3. How often do you visualize data
• Never
• Sometimes – I have created some for school or work
• Always – It's my job

Figure 5.2: Overview of the interview procedure.

or Information Science students, 4 were computing professionals—3 developers, 1 project manager in an IT department) and 8 had other occupations. Participants ranged in age from 18 to 77 years old, with the mean age being 30.7 years old.

We did not conduct a visualization literacy test but asked participants how frequently they visualized data. Most (21) participants reported "sometimes" and seven reported "always." However, in subsequent discussions, we discovered a wide interpretation of "visualizing data" from imagining data to varied frequencies of plotting. We further discuss these results in the analysis.

5.3.2 Setup and Materials

All sessions were conducted through video-conferencing software. Participants were instructed to bring a pen or pencil and a sheet of printer paper to the virtual

Table 5.1: Participants

Participant	Occupation	Dataset
006	Student (CS + Math minor)	File System
007	Student (CS + Management and Info. Systems minor)	Junk Drawer
008	Student (CS + Math minor)	Power Station
009	Student (CS), Software Developer	File System
010	Student (CS)	Junk Drawer
011	Student (CS)	Power Station
012	Student (CS, Biochemistry), CS Teaching Assistant	File System
013	Student (CS, Information Science)	Junk Drawer
014	Student (CS, Information Science)	Power Station
015	Student (CS)	File System
016	Sales	Junk Drawer
017	Substance Abuse Counselor, Swim Instructor	Power Station
018	Web Developer	File System
019	Editor (Retired)	Junk Drawer
020	Software Engineer	Power Station
021	Project Manager (IT)	File System
022	Student (CS)	Junk Drawer
023	Nurse	Power Station
024	Student (CS), Research Assistant	Power Station
025	Student (CS), Research Assistant	Junk Drawer
026	Research Analyst	Power Station
027	Student (CS), Research Assistant	File System
028	Student (CS)	Junk Drawer
029	Student (Chemical Engineering, Information Science)	Power Station
030	Data Scientist, Programmer	File System
031	Army Wife	Junk Drawer
032	Student (Medicine)	Power Station
033	Financial Consultant	File System

meeting, although seven participants used lined paper and three participants used some form of electronic drawing software (e.g., tablet). Each participant was asked to angle their camera toward the paper as they sketched. At the end of the session, participants were told to take a digital (phone) photograph of their sketch and submit it. Sessions typically lasted around 20 minutes, lasting no longer than 30 minutes.

5.3.3 Datasets

We created three (3) datasets which we refer to as File System, Junk Drawer, and Power Station. These names were not shared with the participants. Our goal was to design datasets that afforded multiple data abstractions, based on prior research exploring the facility of changing the data type of an existing data abstraction [13].

We created relatively elementary and sparse datasets with the intention of (1) being accessible to people with a broad range of backgrounds, (2) allowing wide interpretations if they existed, and (3) limiting the need for revising the drawing and thus increasing the likelihood we were observing the initial mental model. We recognize that many datasets are often provided to visualization designers and collaborators "as-is". However, we see the value in *discovering, capturing, curating, designing, and creating* [102] the data and wanted to understand if and how our participants explore data abstractions, in this case, for example, the dataset itself is under construction and thus in flux.

We prioritized keeping the datasets short and understandable, though not necessarily comprehensive. All three datasets were presented in paragraph form rather than as a table so as not to influence the mental models toward tables [10].

We chose not to include tasks with our datasets. In visualization design, tasks are often unclear from the beginning, so in addition to using paragraph form, we provided no additional purpose or tasks to the participants so as not to further influence toward a particular data abstraction.

We discuss the limitations of our choices in the use of paragraphs and the omission of tasks in subsection 5.5.2.

File system

You have two folders. In the first folder are 2 text files and 3 images. In the second folder are 4 text files, 2 code files, and 1 folder. In this folder are 1 text file and 1 image.

The file system dataset was inspired by discussions of file system formats [45] and research regarding difficulties first-year computer science students have with navigating file systems [29]. We collected the age of the participant to see whether we would also notice this phenomenon. We made this a hierarchical set of files to see how participants handled the nested folder. The resulting sketches are shown in Figure 5.3.

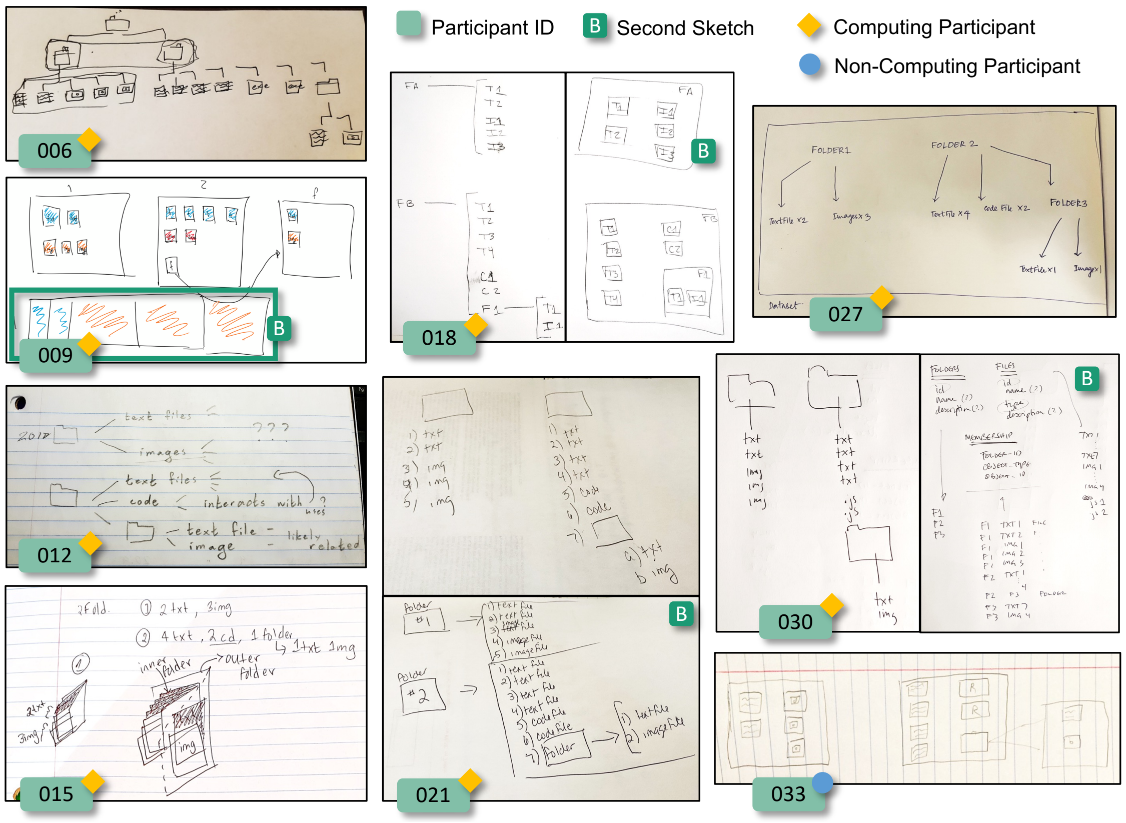


Figure 5.3: Sketches that participants made of the File System dataset. Large versions are in the supplemental archive.

Junk drawer

You have 6 rubber bands, 4 tacks, 3 unused envelopes, a roll of stamps, 4 pens, 3 pencils, 2 sharpies, a small basket, a pencil pouch, and a long plastic basket.

We designed this dataset to be lacking an obvious (non-list) structure, but with several options for imposing one. We included a possible "container" for different groups of items, e.g., the pencils could go in the pencil pouch. We were curious to see if the participants extracted sets or groups from the dataset and how these sets might differ. The resulting sketches are shown in Figure 5.4.

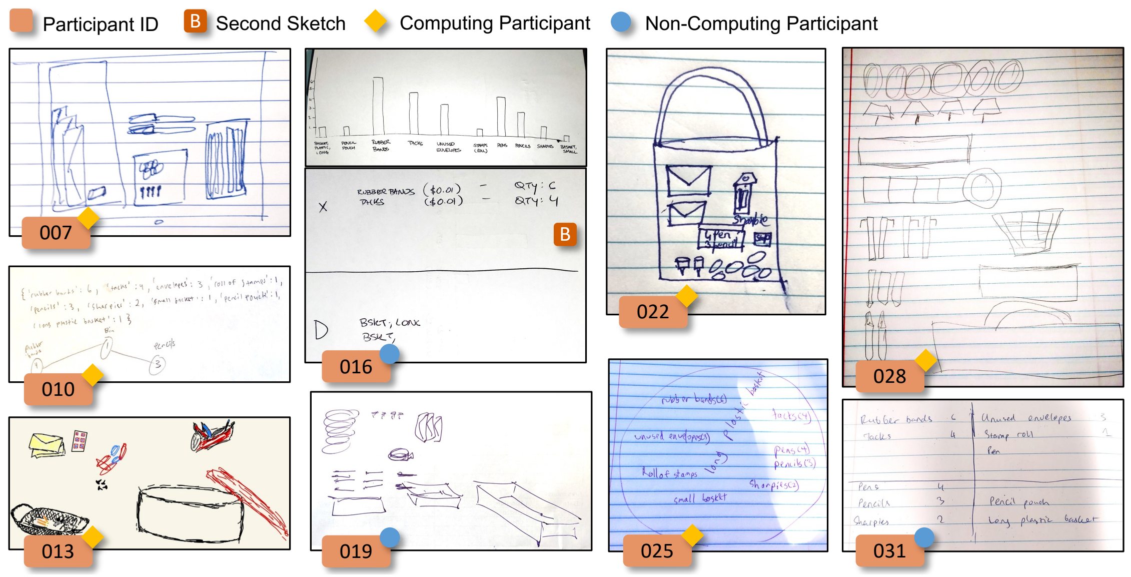


Figure 5.4: Sketches that participants made of the Junk Drawer dataset. Larger versions are in the supplemental archive.

Power station

There are 6 power stations, labeled A through F. Power station A powers 100 homes. Power station B powers 150 homes. Power station C powers 1 warehouse and 100 homes. Power station D powers 4 apartments, each housing 100 residents. Power station E powers 50 homes and 2 apartments, each housing 100 residents. Power station F powers 50 homes.

For the power station dataset, we wanted a variety of classes of data items to allow for different mark types or icons. We also were curious if participants would tie in geographic attributes to the dataset or if we would see any networks, allowing for different visualizations from the previous two datasets. The resulting sketches are shown in Figure 5.5.

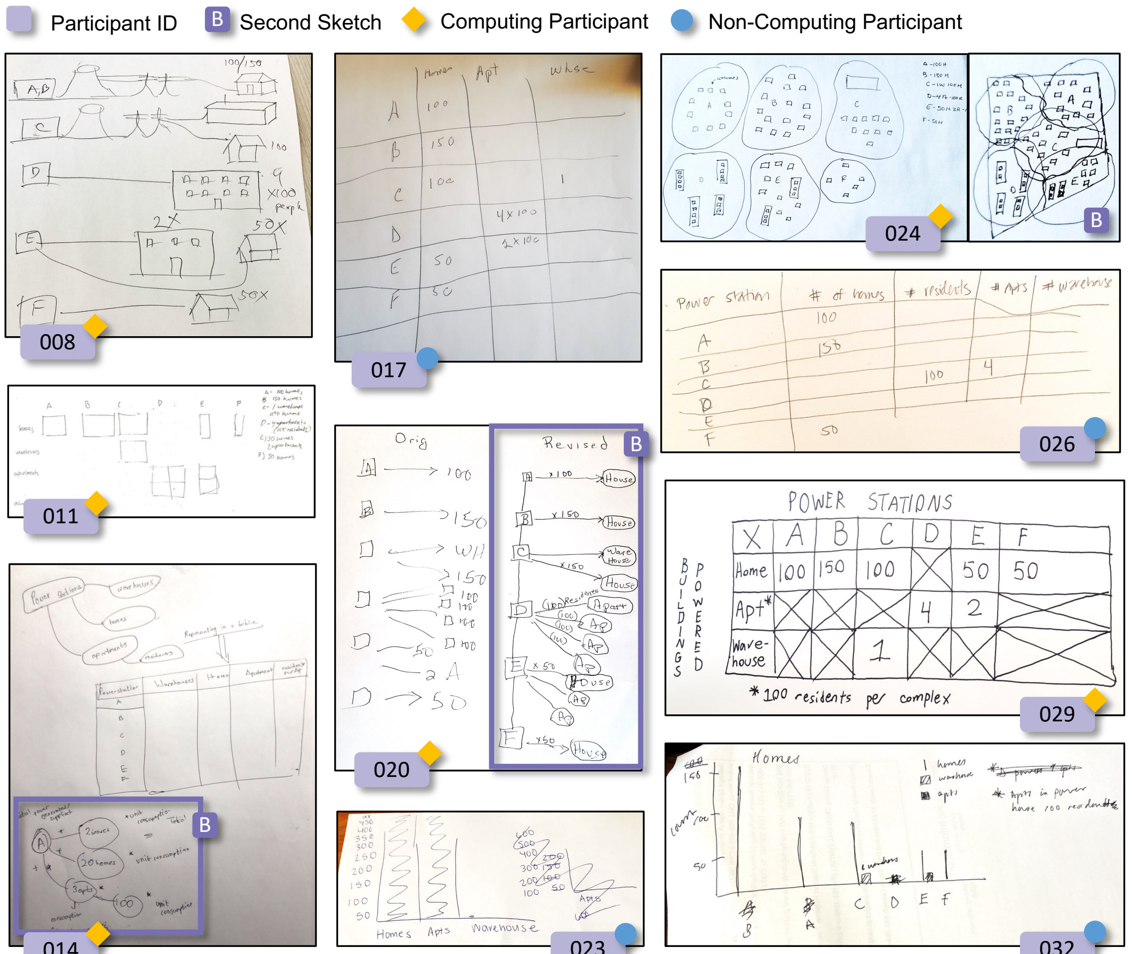


Figure 5.5: Sketches that participants made of the Power Station dataset. Larger versions are in the supplemental archive.

5.3.4 Procedure

We first briefed participants and obtained the study and recording consent. Each participant was then given an overview of the sketching activity verbally and the text of one dataset through the videoconferencing application's chat feature. Our name for the dataset was not included. See Figure 5.2 for the overview script. Participants independently read and considered the dataset, then informed the facilitator once they were through. The approximate time most participants took to read and consider the dataset was under 30 seconds. After this, the facilitator asked, *"What* *was your gut reaction or intuition about the dataset?"*

After the ensuing discussion, the participant was asked to angle their camera and sketch the dataset. Participants were allowed to draw until they felt satisfied with their drawing, with most participants completing their sketches in under 4 minutes. [4] We then conducted a semi-structured interview with the following pre-set questions:

1. Explain the drawing as if I (the researcher) haven't seen it before.
2. What sticks out to you in this dataset?
3. How did you come up with this idea? Have you seen something like this before or have you worked with a dataset like this before?
4. Did your mental model of the data change throughout the drawing process? If so, how?

The first question (explanation of the drawing) is designed to help disambiguate the sketched representation as the authors might interpret it from the participant's view of the sketch. The intent is to separate the sketched visual form from the data abstraction that matches the participant's mental model, providing a way for participants to clarify their representation of their mental model when inhibited by their sketching capabilities. The combination of drawing and interviews is a technique used in mental model research [71].

The second and third questions probe possible influences. The fourth question is designed to provide insight into the possible evolution of mental models, both during initial formation and possibly due to the study design.

In these discussions, some participants augmented their responses by making a second sketch, sometimes prompted by the interviewer to better understand their words. These bonus sketches occurred in eight of the 28 sessions, bringing the total sketch count to 36 sketches.

[4]Participant 013 continued to draw and add detail to their sketch for 11 minutes, at which point the facilitator asked them to stop so that they had time for the discussion questions.

We concluded with demographic questions and a short debriefing. The procedure was designed to take no longer than 30 minutes. Due to a logging error, exact times are missing for five participants, though all finished within their 30-minute slot. The remaining participants finished within 15-25 minutes with a median finish time of 20 minutes. Participants were compensated with their choice of plush toys, a $10 gift card, or a $10 donation.

5.3.5 Thematic Analysis

We took an inductive thematic analysis approach. During data collection, three authors individually noted *codes* and thoughts regarding the transcripts and sketches, initially following an unconstrained *open coding* [101] practice. These codes were recorded as *memos* on a shared GitHub repository[5] to facilitate remote collaboration and to track the provenance of codes.

Though we could have chosen a deductive coding approach for the data abstraction using an existing typology, we deliberately chose to exclusively use inductive coding to not limit, bias, or constrain the data abstractions discovered or our interpretation of the ways the participants spoke about their mental models.

The authors met regularly to discuss the codes, limiting the discussions to the sessions where all authors had had a chance to code. Typically 3-5 sessions occurred between each meeting to discuss initial codes. In total, we coded 28 transcripts and 36 sketches.

As these discussions took place, we moved to axial coding to develop hierarchical concepts. The authors used Google Jamboard to cluster, merge, and split their initial codes and to identify concepts arising from multiple codes. We arrived at 24 consensus codes. The identified concept groupings were then discussed, distilled, and refined into the shared document in the GitHub repository. This permitted asynchronous discussions regarding concepts as they progressed.

The discussions and refinement of concepts led to the discovery of common observations that reinforced codes and also helped us refine our data collection. For

[5]`https://github.com/kawilliams/mental-models-codes/blob/main/codes.md`

example, after observing that participants tended to draw items in the order of reading, we wondered if the alphabetical order of the power station dataset might be influencing this phenomenon. Thus, in sessions 026 and 029, we presented the power stations in non-alphabetical order; however, the participants continued to fill in the data in the order of reading.

The authors initially developed a set of themes from research questions and hierarchical groupings of codes. After external feedback, two authors reconsidered the codes and initial themes, determining the themes had become too broad. Codes and concepts were then reorganized into six themes (described below in section 5.4) elaborating on the research questions and one secondary theme regarding perceptions of data.

5.4 Themes and Codes

We arrive at three clusters of themes relating to mental models of data: mental model content, mental model elicitation, and mental model formation as well as a secondary theme regarding beliefs about data. Below, we explore the themes in each cluster in the context of our study and explain select codes that made up these themes. For detailed supporting evidence for each theme and code, see the supplemental archive.

After presenting our main and secondary themes, we follow up with a discussion of themes regarding our computing and non-computing populations (subsection 5.4.5) and our mental model characterization in discussion with the model of Walny et al. [151] (subsection 5.4.6).

5.4.1 Themes about Mental Model Content

During our thematic analysis, we developed two themes regarding mental model content. The codes comprising these themes have to do with the breadth and composition of mental models. While this cluster contains our best effort in understanding the form of the participants' mental models, it does not contain codes relating to how participants depicted or otherwise communicated that mental model. We discuss those latter codes in the cluster Themes about Mental Model Elicitation (see subsection 5.4.2). The list of codes relating to mental model content can be found in Table 5.2 with the code label listed as "(C#)"; the complete list of corresponding codes and their definitions and supporting data can be found at `https://osf.io/kvnb9/` and in the supplemental material.

Theme	Code	Representative example/Evidence
Diversity of mental models	Diversity of abstractions and representations (C1)	For the power station data, we saw tables (4), set/geospatial (1), bar charts (2), node-link networks (2), set (1), table and node-link (1), and a multi-figure "journal paper"-like representation (1) that included captions and text.
	Ordering diverse, personal (C2) * Caveat: participants drew in order of reading (C3)	Participant 016 (JD) organized by desired category of "durability" based on personal experience.
	Diversity of groupings (C4)	Grouped by functionality (2), participant-selected category (2), grouped only the writing implements (2), list order (i.e., no grouping) (4).
Components of mental models	Physical objects represent data (C5)	Participant 008 (PS): "I've seen a lot of power plants back home... that's why I drew the cooling towers." Participant 013 (JD): "There's a red pencil case that I had during my last year of high school and these are the pens that I have right now in college."
	Tree/Network/Set ambiguity (C6)	Participant 018 (FS) used terms "set" but also "level" and "nesting." Drew a node-link initially but said they considered a nested drawing (shown in bonus for 018).
	Mental models include affordances (C7)	Participant 022 (JD) drew a basket with a handle "so it's organized in a way and you can carry it around." 5/9 participants who had the FS dataset spoke about interactions.

Table 5.2: *Themes about Mental Model Content.* This table contains our themes and codes about mental model content and some representative examples for each code. The codes are labeled as "C#". The complete list of codes and all supporting evidence can be found at `https://osf.io/kvnb9/` and in the supplemental material.

Theme: Diversity of mental models

Across each of the three datasets, the participants chose different abstractions and representations. We classified both the data abstraction (e.g., hierarchy) and the representation used (e.g., node-link) for all sketches. While it is hard to disassociate the typology from the representation in some cases, we took a best-effort approach based on both the sketch and the way the participant spoke about the sketch and their mental model. This led us to classify some mental models as multiple concepts. Figure 5.6 shows our mental model classifications for all three datasets.

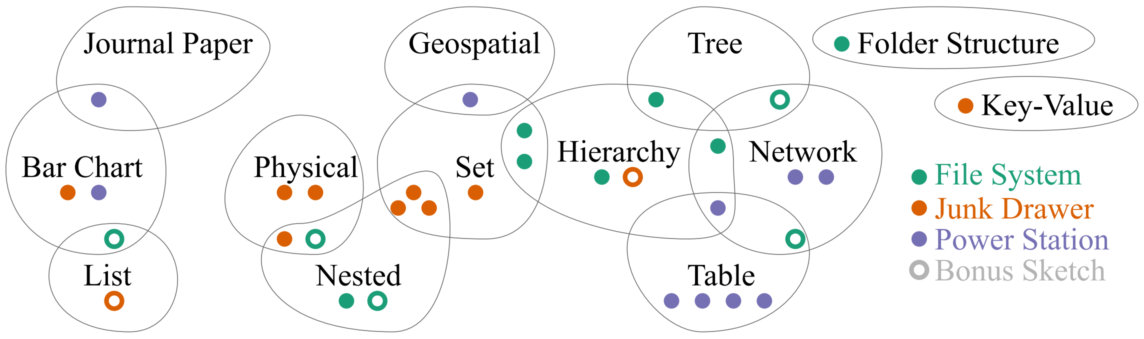


Figure 5.6: Our best-effort classification of mental models expressed by participants. Open circles indicate the second sketch made. Participants expressed a variety of mental models, many of which were ambiguous between multiple categories. Some mental models aligned well with data typologies, while others, like "Journal Paper", did not.

Within the same dataset, we further observed diverse groupings and orderings of the data that had personal meaning to the participant. Participants grouped the data by type, logical association, size, function, and even by the attribute "price" that the participant added based on personal experience. For the file system dataset, participants expressed a desire to reorganize the folders to homogenize file types. The junk drawer dataset was often organized by functionality, by logical associations (e.g., writing implements in the pencil pouch), or by a participant-selected category (e.g., Participant 016 organized by the "durability" of the items, recognizing disposable items might be less valuable). Participants frequently explained their reasoning for grouping the data, with less explanation for the logical and functional groupings in the junk drawer dataset and more explanation for the desire to modify the file system dataset structure, often hypothesizing about reasons for the existing file structure. No two participants grouped their junk drawer items in the same way, except for the no-grouping list order.

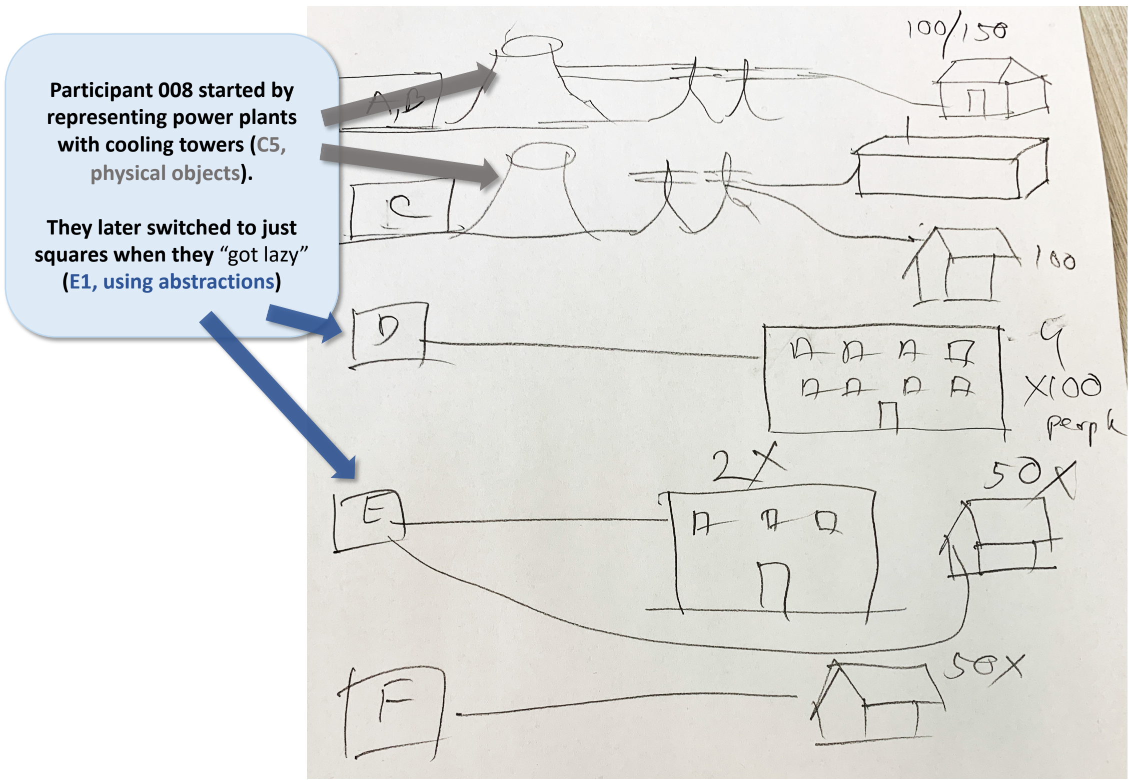

Participant 008 started by representing power plants with cooling towers (C5, physical objects).
They later switched to just squares when they "got lazy" (E1, using abstractions)
A,B
C
D
E
F
100/150
100
9 X100 people
2X
50X
50X

Figure 5.7: The sketch by Participant 008, with annotations explaining how this sketch exemplifies our codes about physical objects and about using abstractions (C5 and E1, respectively).

One caveat to the ordering: despite the different orderings we observed, most participants still drew the data in the order of reading. We observed 22 participants draw their dataset in the order in which they read the dataset, and 4 participants draw the dataset in a way that did not reflect the order presented in the dataset (2 participants were not able to easily display their sketch to the camera while drawing, so we did not consider their sessions for this code). All participants who had the file system dataset drew it in read-order. Most participants who had the junk drawer drew in read-order (7/9 participants), and most participants with the power station dataset drew in read-order (9/11 participants). Those 4 drawings that were not in read-order were drawn in order of some internal mental grouping or categorization: the 2 participants who had the junk drawer dataset discussed logically grouping the items, the 2 participants with the power station dataset drew representations for the categories of power (home, apartments, warehouse) rather than sketching a representation of the first power station and its recipients.

Theme: Components of mental models

We developed three codes regarding the components of mental models, in particular, regarding the presence of physical objects, ambiguity in mental models involving relations such as trees and sets, and the presence of affordances.

Physical objects were prevalent in the mental models we observed. The drawings of the objects mirrored their appearance, affordances, and orientations in the real world. This theme cross-references codes under Mental Model Content and Mental Model Elicitation, but as these physical objects were what the participant thought of as their mental model, we placed this code under Mental Model Content. The appearances of the objects were tied to memories: the cooling towers drawn by Participant 008 were based on power stations the participant had seen in their hometown and engineering textbooks, while nearly all of the junk drawer items drawn by Participant 013 had a story or memory tied to them. We note the strong semantic connection between our datasets and concrete objects may have influenced these observations and discuss this further in our Limitations section (subsection 5.5.2).

We found difficulty disambiguating and naming mental models that involved relationships between data items. For example, mental models similar to data abstractions typically classified as trees, hierarchies, and sets. We carefully considered language cues—terms such as “levels,” “branches,” “associations,” “nesting,” “underneath,” “inside,” “hierarchy,” and “graph.” There were no clear boundaries in how representations were used, and sometimes the terms vocalized were associated with multiple different data abstractions. For example, Participant 018 used the term “set” and the term “level” in describing their node-link sketch.

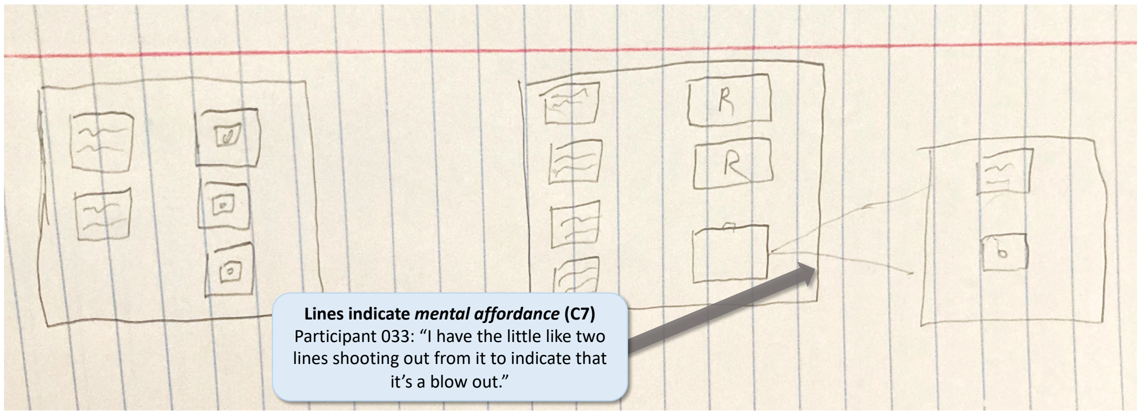

Lines indicate *mental affordance* (C7)
Participant 033: "I have the little like two lines shooting out from it to indicate that it's a blow out."
R
R

Figure 5.8: The sketch by Participant 033 of the file system dataset. The participant drew lines to indicate interaction in their mental model (code C7).

Some participants described interactions, or *mental affordances* [102], within their mental model. Six participants who had the file system dataset explained how they would interact with it, specifically how they would navigate it (Participants 006 and 030), or even actually drawing an inset to show this interaction (Participant 033). Even though we were discussing an abstract concept (i.e., their mental model) in a static medium (i.e.,paper), participants referred to interactions with their visualization and mental affordances, or internal interactions, that they used with their mental model.

5.4.2 Themes about Mental Model Elicitation

We developed two themes regarding mental model elicitation, encompassing how the mental models were drawn on paper and how they were verbally described by participants. These codes solely relate to the choice of representation and encodings on the paper and what verbiage the participant used to describe their drawing. The list of codes relating to mental model elicitation can be found in Table 5.3 with the code label listed as "(E#)"; the complete list of corresponding codes, their definitions, and backing data can be found at `https://osf.io/kvnb9/` and in the supplemental material.

Theme: Depictions of mental models

We formed several codes regarding how participants depicted their mental models, such as their use of text, legends, details, and abstractions, as well as where they were constrained by the sketch format.

We only noted the use of text when the participant commented on their use of text. Participants specified that they would use words to communicate with another person (Participant 019, 027), with Participant 027 noting, “When I use icons, unless it’s mutually understood by both people, it might confuse; or even I might forget what the notation actually stood for... You can’t go wrong with text, and it’s [the file extension] not long either.” To help their understanding of the file system dataset, Participant 021 decided to put “2 code [files]” since they did not know what code meant. Other participants chose a code type for clarity—Participant 009 used “.java” so that “we can be more explicit” and Participant 030 used “common file extensions” but recognized they used a mix of “tokens” for the file types.

Similarly, the use of detail was only noted if the participant commented on adding detail. Some participants wished to add detail to distinguish between the junk drawer items (Participants 019, 031). Other participants wanted to add arrows and labels: Participant 012 added labels and arrows to suggest hypotheses about the relationships between the files, Participant 020 wanted to add weights to directed arrows for the power stations and said that the addition “would be an improvement.”

Only one participant drew a legend on their sketch (Participant 032). Other participants verbally described what the icons meant, like explaining the icons for the types of files (Participant 006) or explaining that the small squares represent homes (Participant 024).

Some participants either vocalized their use of an abstract mark or switched to a more simple mark during sketching. Participant 011 drew boxes because “drawing houses would be too difficult;” Participant 014 sketched the idea of a table rather than the full one; Participant 024 said their mental model was geographical but chose to draw without geographical marks. Participant 008 started drawing buildings in 3D, but then switched to 2D icons. To show their reaction to the file system data, Participant 012 said “I marked it with a bunch of question marks to the right because I don’t have any idea what [this folder] was for; it’s just there.”

Other participants left out details altogether. Participant 009 originally labeled the text files with “txt” but stopped labeling them because “I’m gonna be lazy.” In their bonus drawing, Participant 030 added an ellipsis for the “OBJECT_TYPE” attribute after writing one complete data table entry since the rows beneath were all the same type. Participant 023 used a squiggly line instead of a rectangle for the bar graph; the lack of detail is possibly related to their level of math literacy (code F7).

Some participants ran out of space while drawing and verbally noted it. To adapt, some added their marks to a different location (Participant 007 drew the sharpies outside of the pencil pouch, and Participant 016 skipped back to the left side of their x-axis since they ran out of space going left to right). Others continued with the existing drawing and expressed regret (Participant 015 said “it’s hard to draw this. I should’ve brought a pencil.” Participant 022 wished they “made the basket a little bigger.”).

Some participants were constrained by the encoding schema they chose, rather than space. This happened to Participants 030 and 033 when they encountered the code files in the file system dataset since they did not anticipate the file type and had not prepared a way to encode it.

Theme: Communication with others

We observed participants making conscious choices about how they represented the data when communicating with others (code E6[6]), but varied in their level of detail and abstraction, and use of terms.

Some participants sketched at a high-level abstraction but added detail, like

[6]The complete list of codes can be found at `https://osf.io/kvnb9/` and in the supplementary material.

colors (Participant 009) or item detail (Participant 031), to clarify for others. One participant re-oriented their tree from top-down to left-right and attempted to add interaction indications (Participant 021). Participant 031 recognized which aspects of their drawings could be confusing and said, “Between the pens and the pencils and the Sharpies, you can’t really tell what they are. If I were to give it to somebody, they probably wouldn’t be able to tell—to differentiate between those groups...I probably should have written ‘envelopes’ on them or some type of—you know, if somebody were to look at this, I don’t think they would know what I drew.”

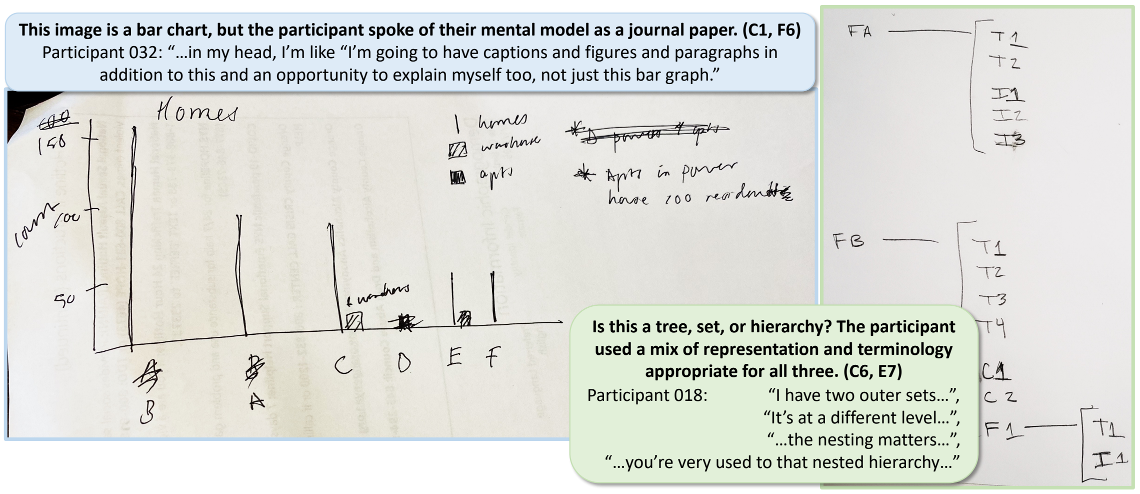
Two participants said they would choose a different data abstraction, depending on the audience’s “quantitative literacy” (Participant 026), or they would find a “better way” to represent the data, possibly by adding a table or other figures and captions (Participant 032).

When communicating with the facilitator, participants added annotations when discussing their sketches. Participant 006 added encompassing circles around the top-level folder of their file tree and the children under folder 1. To explain how they would solve for the total power generated, Participant 014 added a graph with root node ‘A’ at the bottom of the page.

Sometimes participants used terminology in conflict with visualization community concepts for dataset abstractions. One participant drew a table, even though their description and interaction with the dataset focused more on data item relations (Participant 014), reinforcing the code about ambiguity when using trees/networks/sets from Table 5.2. In particular, they used the terms “endpoint”, “layers”, and “map” and relied on their other drawing of a node-link graph to augment their description of how they would solve for the amount of power produced. When referring only to the data (no longer problem-solving), they said what stuck out to them was the “layers” and “sublayers” in the dataset.

One participant used set-like terminology to describe their node-link diagrams. Participant 006 described, “In my head, I’m oddly enough in the folder that those two folders are within,” and often used “within” and “in” to describe the location.

In response to the interview prompt “Describe your sketch”, we observed a range



This image is a bar chart, but the participant spoke of their mental model as a journal paper. (C1, F6)
Participant 032: "…in my head, I'm like "I'm going to have captions and figures and paragraphs in addition to this and an opportunity to explain myself too, not just this bar graph."

Is this a tree, set, or hierarchy? The participant used a mix of representation and terminology appropriate for all three. (C6, E7)
Participant 018: "I have two outer sets…", "It's at a different level…", "…the nesting matters…", "…you're very used to that nested hierarchy…"

Figure 5.9: The sketches by Participants 018 and 032. Annotations explain how the verbal description from Participant 032 expanded their representation from solely the bar chart that they sketched to an elaborate multi-figure "journal paper" (C1, F6). The annotation to Participant 018's sketch includes quotes that highlight the mix of terminology that the participant used (codes C6, E7).

in the level of descriptive detail. We categorized the levels of detail in the verbal descriptions : (1) individual data points, (2) individual icons, (3) relations of icons or positions of icons, and (4) data abstraction. By "individual data points," we mean the participant nearly restated the dataset and did not describe the drawing. Five participants stuck to this individual data point level of detail (Participants 008, 014, 018, 027, 032).

The next level of detail, "individual icons," means the participant gave visual descriptions of the icons or marks used in the dataset. These verbal descriptions ranged in detail, with six participants matching this level (Participants 010, 011, 013, 019, 030, 031). Some participants named every type of mark, while others got distracted midway.

The third level of detail, "relations of icons or positions of icons," means the participant stated where the icon was on the page or in relation to other icons (e.g., "in a folder," "next to the files," "roll of stamps down there and tacks to the right"). Nine participants referred to relation/positioning when describing their sketch (Participants 006, 007, 012, 015, 021, 022, 024, 025, 033).

Eight participants named a data abstraction (Participants 009, 016, 017, 020, 023, 026, 028, 029). However, their name did not always match the visualization community's name for data abstraction or data representation that they used. After naming the data abstraction or representation, they went on to describe the icons or markings they used, the second level of detail.

5.4.3 Themes about Mental Model Formation

We describe two themes describing our observations regarding how participants came to their mental model, based on their descriptions. These codes do not attempt to explain how the mental models are formed; instead, they are observations of how a mental model develops in a data- and visualization-related setting. The list of codes relating to mental model formation can be found in Table 5.4 with the code label listed as "(F#)"; the complete list of corresponding codes, definitions, and supporting data can be found at `https://osf.io/kvnb9/` and in the supplemental material.

Mental model formation process

Participants suggested their mental models form quickly, with little change, though we observed they became more detailed during the session. Our first interview question asked about initial impressions and reactions. Participants' responses already expressed ideas for data abstractions and what they would draw, including thoughts about ordering and purpose-seeking. Participant 024 said, "My gut reaction was like an image of—I dunno if you know cell-free MIMO graphs..." and described how they would use the idea to draw the power stations. Other participants immediately tried to find the purpose or context of the dataset, such as supposing that the junk drawer dataset "it's like a handy toolbox for a home" (Participant 013). Participants also expressed a desire to organize the data by categorizing or by finding a more "efficient" way.

We later asked participants if their mental models had changed. About 60%

(17/28) of participants said it did not. Several participants mentioned aspects of their mental model that were obvious, such as “it’s obviously a folder structure” (Participant 009) or “I feel like in my head it’s the simplest conclusion” (Participant 032). For the participants whose mental models had significantly changed, they often cited trying to find a “better” or “best” way to display the data (Participants 010, 016, 023, and 032).

We observed some participants vocalizing their revisions, adding details to their sketches and mental models as they drew, or adding clarifying information afterward when describing how their mental model changed throughout the interview. Participants 030 and 033 chose icons to represent different file types during the drawing. Participant 030 explained when asked about mental model changes that this was a “minor hiccup... trying to choose tokens to represent the category of files: text, image, code.” While explaining their sketch, Participant 006 added a root to their file system.

Influences on the mental model formation

When asked where their idea for their mental model originated, many participants explained where they had seen something similar or hypothesized the source of their inspiration. All but one participant who drew the file system dataset cited an operating system or software for either the structure or icons, including Participant 015 who drew the nested manila folders similar to a Windows icon. Despite the common inspiration, there were several different types of depictions. Across all datasets, participants cited a real-life example (e.g., a drawer in their home, power plants, cell-free MIMO) or their work as the reason for the choice of data abstraction.

Participants with less math literacy had a limited representation and mental model. Two participants expressed math hesitancy, 023, “Yeah, that’s a lot of math. I’m not - I’ve only taken high school math”, and 031 “I am not very good in math.” Both were part of our non-computing population of participants. Participant 023 had difficulty finding a concise way to express the power station dataset. Their sketch was missing a data dimension (the power stations A-F). Participant 031 had the junk drawer data and drew the physical objects as given. While the evidence for this code is lighter as we suspect our IRB-approved advertisements may have dissuaded people not comfortable with data (see Limitations, subsection 5.5.2), we noted this code for future investigation.

Since we only provided a dataset to the participants, participants often wanted to add additional data or information to the dataset or they added additional context or a hypothetical source for the dataset. We classify such behavior as *purpose-seeking* behavior, or an attempt to connect with the dataset in an imagined real-life setting. Six participants suggested other data item attributes they would want, and four participants added relationships between items. This was most prevalent in the power stations dataset with requests for the number of people per house, power requirements per building, and one instance of geospatial coordinates. In the file system, additional data requests manifested as the wish for file sizes, code file types, and suggested relationships beyond the folder structure. In the junk drawer, price, durability, and nesting structures were suggested or imposed.

Participants also suggested a task associated with the data. With the power stations, three participants wanted to solve which station produced the most or least energy. Other assumed tasks included taking inventory (Participant 016), cleaning (Participant 019), carrying (Participant 022), determining affected people or buildings (Participant 029), solving a math problem (Participants 014, 031) or presenting in a scientific journal (Participant 032).

Half of the participants (14/28) contextualized their dataset with a suggested source or generator of the data. The junk drawer dataset was suggested to be a "stationary drawer" (1 participant), "office supplies" (4 participants), and "an electrician's toolbox" (1 participant). Participants who received the file system dataset imagined a new program or provided reasons why a program would be organized in the given manner. Participants with the power station dataset supposed that the data was for a "residential part of the city" (2 participants) or a "municipal or power company guide" (1 participant).

5.4.4 Beliefs about Data

In addition to our main themes regarding the content, elicitation, and formation of mental models, we developed a secondary theme describing our observations about participants' beliefs about data and data analysis.

The idea that data relates to tables is prevalent (code D1). Four participants who did not use tables mentioned expectations involving tables. The facilitator asked Participant 007 if they were surprised that the junk drawer dataset was given as a dataset and they replied, "I was expecting something more structured, maybe like a table or something. I guess I was expecting a table. Because that's the most common form of storing data, like a spreadsheet or table, something like that." Participant 019 also received the junk drawer dataset and said, "I didn't work with Excel very much, so I don't think of datasets, but when I heard the term 'datasets', I really thought about the analysts I worked with and Excel data, and I thought of big datasets and grouping people by demographics, that kind of thing. I refer to datasets and I was familiar with them but I never thought of stuff like this as a dataset."

Two other participants mentioned using tables to organize the data via relational tables (Participant 030, bonus sketch) or to answer questions about the data (Participant 014).

There was hesitation regarding whether the given dataset was truly data (code D2). Initial impressions of the dataset included impressions on the term "data" itself. Participant 007 concluded, "I guess this [the junk drawer dataset] is a valid dataset, it's got objects and quantities for those objects." Participant 010 associated the word "data" with relating to computers, and drew a Python-like dictionary of the junk drawer dataset.

When asked about how often they visualize data, Participant 012 considered that it "depends on what you consider data." Participant 019 responded to this question by relating data analysis to grouping people by demographics: "I thought of big datasets and grouping people by demographics, that kind of thing."

Two participants made us question the distinction between the dataset and the data. Participant 014 distinguished the dataset and the data within it as different ideas: "like the emphasis should be on the dataset, not the data containing [sic] in it, right?" Participant 027 didn't think of the items in the file system dataset as the data; instead, they assumed the data was inside those items (files) and not explicitly given.

5.4.5 Differences between computing and non-computing participants

We recognize that the majority of our participants had a computing background. While not part of our original experiment design, we revisited our codes to check if any were heavily computing-biased in their evidence. Of our codes, the following codes had solely computing-based evidence:

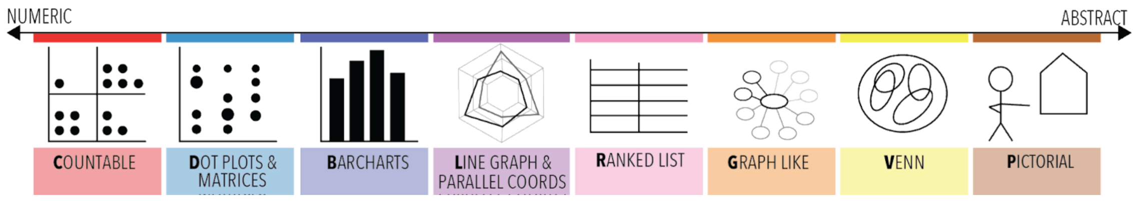
- Node-link sketched representations (all 9 node-link sketches were done by computing participants– discussed in subsection 5.4.6),
- E1 (from 5 computing-related participants and 0 non-computing-related participants): using abstractions in the depiction,
- F5 (4 computing, 0 non-computing): purpose-seeking by adding relationships between items, and
- C5 (3 computing, 0 non-computing): using physical objects that were cited from prior experience to represent data.

The code about math literacy (F7) was from 2 non-computing participants. All other codes contained supporting evidence from both non-computing and computing participants. See subsection 5.5.2 for more discussion on our participants' relationships to data.

5.4.6 Comparison to prior work on sketching and data reports

Given the similarities between our study and Walny et al.'s sketching study [151], we examined our sketches on their numeracy-to-abstractness continuum. Figure 5.10

shows our best-effort categorization, given that we used a different dataset and therefore saw different data representations than Walny et al. (their dataset was a table of behaviors in social scenarios). The session number for each sketch is placed within the data representation category along the continuum. Possibly due to the nature of our experimental datasets and population, we did not observe any sketches in the “line graphs and parallel coordinates” category, nor any in the “ranked lists” category. Of the 28 sketches (not including bonus sketches), 8 leaned toward the numeric side of the continuum and 20 leaned toward the abstract side. The category with the largest number of sketches for our participants is node-link representations, whereas the most common representation for Walny et al. was bar charts. This effect is likely due to the qualities of the datasets given to the participants.



Numerica/Countable/Table	Table with symbols	Bar charts	Line Graph & Parallel Coords	Ranked List	Node link / NL hierarchy	Containment / Sets/ Venn/ hierarchy via enclosure	Pictorial / Physical (with and without nesting)
🟨 10	🟨 11	🟦 16			🟨 6	🟨 9	🟨 7
🟦 17		🟦 23			🟨 8	🟨 24	🟨 13
🟦 26		🟦 32			🟨 12	🟨 25	🟨 15
🟦 29					🟨 14	🟨 28	🟦 19
					🟨 18	🟦 33	🟨 22
					🟨 20		🟦 31
					🟨 21		
					🟨 27		
					🟨 30		

Figure 5.10: Our categorization of data representations from the participants’ sketches, placed along the numeracy-to-abstractness continuum of Walny et al. [151]. Yellow shading indicates the participant has a computing background, blue shading indicates a non-computing background.

Within each category of abstraction, there was a mix of computing and non-computing participants represented, except for the “node-link/node-link hierarchy” category (9 computing participants, 0 non-computing), “bar charts” (0 computing participants, 3 non-computing participants), and "table with symbols" (1 computing participant, 0 non-computing participants). The prevalence of computing participants in the node-link category can partly be explained by the dataset they received: 6 of the 9 participants coincidentally received the file system dataset, and the other 3 received the power station dataset. These participants may be more familiar with node-link diagrams, especially related representations commonly used in file systems.

Walny et al. also examined the participants' written reflections about what they had discovered in their datasets, which the authors call *data reports*, and the authors developed a *data reports spectrum*, which placed responses that contained direct readings of individual data values at one end and higher-level conjectures and hypotheses at the opposite end. A major finding from intersecting their participants' sketches with the data reports was how "the participants who submitted the most abstract sketches were among the participants whose data reports tended to be in categories E3 (including extrinsic information) and F (statements with analytic potential)."

To test this finding in our work, one author reviewed the interview transcripts for such statements. The author chose to exclude statements in category E3 because the interview question, "How did you come up with this idea? Have you seen something like this before or have you worked with a dataset like this before?" prompts the participant to relate the data to external information, which would not be an organic source for such statements. Therefore, only F statements, statements that offer fledgling hypotheses or conjectures about reasons for values, were included. For example, an F statement from Participant 008 is, "I mean, I don't know what their fuels are, I'm assuming they're maybe coal-powered power plants." Another example is from Participant 009: "I don't know how big it [the file] is, at the end of the day, right, so the text file could be bigger... could be super big, could be smaller."

We found such F statements (statements with analytic potential) from 14 participants. Of those 14 participants, 11 of the participants' sketches fell on the more abstract, pictorial side of the continuum, supporting Walny et al.'s finding that participants with more abstract sketches included statements that were more analytic on the spectrum of data report statements. Our results thus show reasonable agreement with those of Walny et al., given the differences in study design.

5.4.7 Comparison to prior work on mental models

Our findings bear similarities to the Paris map study of Milgram and Jodelet [99]. In that study, an individual's mental model of their hometown included important locations and roads connecting them, and their personal background influenced the order and size they drew these locations. In our study with datasets, we observed specific data points had personal connections, like the locations in Milgram and Jodelet, but also outliers were of interest. We further observed that people drew from their personal knowledge to communicate "rules" of the dataset, which they utilized to determine outliers or suggest new data. For example, some of the File System participants suggested alternate organizations based on file types.

Other prior work evaluated the accuracy of people's mental models when learning new phenomena via text and visual elements, only text, or only visual elements [21, 129, 58]. However, most datasets start as text-only collections and are frequently not human-readable, so we did not present visuals and provided only text. Datasets may not necessarily have relationships explicit, but rather are for the individual to determine, leading to more ambiguity as we observed.

Across studies, participants choose to emphasize elements and connections in the dataset that they have a special connection to, that reflect aspects of their demographics and background, and draw inspiration for their drawings from other maps and data visualizations that they had seen. Like eliciting the Parisian maps and our dataset sketches, people have a wealth of knowledge and expertise that they may not realize. They have an intuition about the dataset and have an idea of what "makes sense" in the data, even if some of these ideas may be inaccurate. This knowledge may be subconscious yet useful for visualization designers, especially the unwritten "rules" of a dataset. Further work in this area can help visualization designers work more efficiently and produce more useful visualizations for their users.

5.5 Discussion

We discuss our findings regarding the research questions we initially posed. We then discuss the limitations of the study. Finally, we discuss the implications of our findings for visualization design.

5.5.1 Revisiting our Research Questions

Our study was inspired by the research questions listed in section 5.1. We revisit those questions and discuss our discoveries towards them.

What factors influence people's initial mental models of data? We observed that participants quickly came to their mental models, with several (11/28) expressing how they would represent their mental models right after reading the dataset. Most (17/28) remained consistent in the high-level data types they sketched and discussed, though details regarding the particular encodings required further consideration. This consistency leads us to believe that the sketches and descriptions were close representations of the participant's mental model in many cases.

In a few cases, however, a participant realized their mental model required revision. While drawing they realized their approach did not permit them to add all of the data from the paragraph (code E2). Some also expressed the desire to approach it differently after they had completed their first sketch.

When discussing how they arrived at their mental model, many participants (16/28) related their sketch to something they had seen before, some directly applicable to the data, such as operating system file browsers for the file system (7 computing participants, 1 non-computing participant), and some less direct. Two participants discussed recent sources of inspiration such as coursework. We hypothesize that our participants were able to connect their mental models to existing visualizations and data representations because of the accessibility of the datasets and the data-literate background of our participants. Further study is needed on less-accessible datasets and people with low data literacy.

Ideas about data organization and structure also influenced how our participants sketched the datasets. Three participants suggested expectations regarding how “data” should be organized. Four participants suggested their model was obvious.

Our participants’ mental models were also influenced by inferred purposes, which had implications for the corresponding data abstractions. Participants added purpose and context to the dataset by suggesting a generator for the dataset, problems to solve with the data, or insights they wanted to glean from the data (see codes F4, F5, F6). The inferred purposes led them to further suggest more data or attributes that could help achieve these imagined purposes and create more elaborate mental models than ones based strictly on the data alone.

What encodings and visualizations do people commonly use to communicate their mental model? We observed a variety of encodings and visual representations. Tables, node-link diagrams, containment/enclosure, indented nesting, icons/physical depictions, proximity for grouping, and bar charts were each seen multiple times (see Figure 5.10). Beyond bar charts, there were no common statistical charts. This may be an artifact of our dataset design, which was designed to enable the use of one of several data abstractions from data typologies.

The least diverse set came from the file system prompt, where node-link diagrams and enclosure were common approaches, though one participant drew a physical depiction inspired by Windows icons (Participant 015).

Participants used a range of specificity and generality in their marks, for different reasons. Many participants used abstract marks or elided details, some from the beginning of their sketch and some changing to more abstract marks along the way for efficiency during drawing (5 participants remarked when they deliberately made this choice). We saw text used in tables, bar charts, file system icons, and labels, and exclusively text in three of the junk drawer depictions rather than physical icons. Participants explained that their use of text was to clarify (when representations were unclear or they wanted to specify) or to simplify their representations (rather than drawing items). These explanations may indicate that the participants were unconsciously aware of their communication efforts: they used a shorthand version of the representation and skipped drawing details when they felt a viewer could understand what they meant, yet they added text to clarify when they felt a viewer may misunderstand a sketched representation.

How do they describe how they think about the data? How do people describe their sketches? We observed a spectrum in the level of detail participants expressed when describing their sketches. The level of detail ranged from essentially repeating the dataset, to describing a verbal legend of the marks, to naming the data abstraction or representation. This differing specificity has implications for how people communicate about datasets and how they emphasize important aspects of datasets or visualizations. Two people with the same dataset may value different levels of granularity in the data (e.g., one may care about individual data point values while another may only want to see regression lines). A visualization designer must be sensitive to both perspectives and weigh how or if they want to encode the data to support these views.

Participants made deliberate presentation choices with their sketches while presenting their sketches to the facilitator. Some participants added detail either to clarify the depiction for the facilitator or to emphasize parts of the sketch to the facilitator, e.g., circling the part they were explaining. Some participants also suggested changes they would make for another audience, such as re-orienting the sketch to make it easier to read, changing the data abstraction entirely, or adding more explanation or a legend. These changes in presentation pose interesting questions on the transferability of mental models of datasets and how well a person can communicate their mental model to one another to create a shared understanding of the data.

In the post-study demographic questions, participants described how often they visualize data by citing software (e.g., Excel, Tableau, D3), types of charts (e.g., box plots, radar graph, bar graph), but also describing mental imagery (internal visualization) and plain visual representations (e.g., a list, note-taking). The participants who mentioned software described interacting and analyzing data through the use of spreadsheets and statistical software. The participants who interpreted "visualize" to mean "imagine internally" gave more descriptive answers about how they mentally interact with and think about data. These participants described estimations, relationships, and trends in data, and used more inward vocabulary ("organize in my brain", "imagine in my mind").

How difficult is it for people to sketch and/or describe their mental model? How difficult is it for us to understand? Nearly all participants began sketching right after our discussion, with some attempting to draw before the initial discussion question. Participants paused during their sketching when they ran into a space constraint, schema constraint, or an outlier (e.g., the warehouse in the power station dataset), but otherwise, drawing was uninhibited. Occasionally they paused to evaluate their current sketch and then modified it. Often participants paused when asked what sticks out in the dataset, suggesting they were thinking and evaluating the dataset against previous experience or searching for outliers. The participants who quickly expressed an answer to what sticks out had typically mentioned the phenomenon earlier (e.g., how to draw the code files, the structure of the file system, the warehouse or apartments in the power station dataset).

When asked to describe their sketch, depending on the level of detail given, the facilitator asked questions to try to better understand their mental model and to get verbal descriptions of the visual phenomena. Some participants willingly launched into more detail about their sketches with minimal prompting. The participants that were less willing to discuss details of their sketch may have felt their sketch was self-explanatory—a trend with some of the sketches of tables and pictorial sketches of the junk drawer. However, those participants did have ideas about the data itself, hypothesizing about the source or asking for different attributes or more data.

We encountered difficulties in classifying the mental models. There were ambiguities in the terms used. Understanding the relations between data items and the structures that the participants imagined was especially difficult due to the ambiguity of terms. Another ambiguity was that participants suggested their model was obvious without a line of reasoning. Some used language that read the data back rather than describing the data in a new way, whether it was the individual items or more holistic descriptions like "a folder structure."

While we are not confident we fully understood any participant's mental model, or we ever could, the combination of sketching and semi-structured interview did help us gain a significant understanding in a relatively small amount of time. Perhaps a second step where the facilitator and participant draw a second dataset independently, with the facilitator trying to mimic their understanding of the participant's mental model, may serve as validation of our understanding.

5.5.2 Limitations

We discuss the limitations of this study concerning the participant population, the dataset prompts, and the study procedure.

Limitations in participant population

Our participant population was likely biased toward more data-literate people. This is probable for three reasons: (1) all sessions were organized and conducted online over Zoom; (2) the header of the advertisement for the study was "How do you imagine a dataset?", which may have attracted people interested in data; (3) due to the ongoing pandemic, we did not recruit participants in person in common high-traffic areas, aside from a trio of flyers posted at a local YMCA.

In particular, we had a high proportion of participants with computing-related occupations (16 students, 3 professional developers, and 1 manager), which may have influenced the breadth of data abstractions we identified. Computing participants were the sole users of node-link diagrams, though this is in part due to their high concentration among File System participants. However, even with this almost homogeneous set of participants, we observed diversity in their representations, data abstractions, and ways of speaking about the File System dataset. This diversity could be magnified with a more diverse group. Our analysis of data abstractions used across all participants (code C1, figures 5.3, 5.4, 5.5) showed that other than the node-link diagrams, visual forms used by several participants were mixed between computing and not-computing participants.

We also collected the age of our participants to see if there was a relationship between the participant's mental model of the file system dataset and their age. The younger participants tended to be computer science students or similar so they were familiar with file systems. Thus, we saw no relationship between age and understanding of file systems.

Limitations in the study dataset prompts

The datasets we designed do not represent the full spectrum of data we see across the field of visualization. For example, continuous values are not directly represented in these datasets, which focus more on counts and relations. Still, one participant considered geographical location as a missing column in the Power Station dataset and showed how they would include it in their thinking.

All three datasets had data items that have relations to concrete objects (e.g., files and folders, office items, buildings). This was done to make the data more accessible to a wide audience. More abstract items are not handled in this study. We did not include the names of these datasets when sharing them with participants, to avoid further bias. However, the strong semantic meanings of the data items may have influenced our findings in ways that we would not observe with more abstract data.

Regarding the semantic forms associated with the data, some of the authors initially had strong semantic notions, like "a file system is obviously a tree," but informal discussions revealed that these notions are far from universal, which was an impetus for the research questions and later the dataset prompts.

We decided against providing purpose, a context of use, or tasks with the datasets. Our rationale was that early in the design process, these tasks and data are often in flux. However, there is typically *some* notion of purpose for a visualization (e.g., to analyze, compare, or predict) that will influence tasks. Data abstractions are often task-sensitive, so by omitting purpose or a set of tasks, we may have observed a more diverse set of data abstractions than in a purpose-influenced study. We hope purpose and task influence can be further explored from this study's baseline.

Our dataset prompts are small in the number of data items. We chose the small size so participants could consider and sketch the full dataset. When communicating with visualization designers, collaborators often start by offering a toy dataset to aid their explanation, which is closer to the dataset size we use. Further research is needed to understand the strategies people have in forming a mental model of bigger datasets. Thus, our study does not answer questions as to how participants' mental models might change between a toy and a full-size dataset, how they communicate datasets too large to draw, and how they might choose to describe and represent the data, for example in terms of overviews, details, or aggregations.

In addition, the datasets and design of this study do not cover uncertainty in data, dynamic data, or data that necessarily forces multiple abstractions. Therefore, we do not report on these cases but note that even in our simple case, we observed diversity in data abstractions and difficulty in describing relations, thus we would expect more pronounced indications of these phenomena in more complicated data.

We presented the datasets as a text paragraph to avoid adherence to a given data abstraction or representation, such as a table, observed in prior studies [10, 149]. However, we did observe a tendency to observe the listed order in the paragraph rather than re-order the data among some participants, especially in the Junk Drawer dataset. This could indicate the participants were basing their model on the implicit list. This was not universal though, as other participants re-organized the same dataset.

On a more specific note, there was some confusion with the "long plastic basket" in the junk drawer dataset. We envisioned this to be a holder for envelopes or writing implements, but we received many different interpretations that revealed the ambiguity of the word "long" (Participants 013, 019, 022).

Limitations in the study procedure

Our aim with this study was to capture the initial mental models of people regarding data form and abstractions, before being presented with one. We acknowledge that mental models are ever-evolving and that mental model elicitation is difficult [71]. We chose a direct elicitation technique via interviews and drawing as it is as effective as others. Like other mental model elicitation methods, it can only provide a representation of the mental model and is dependent on the participant's drawing and verbal descriptive abilities, as well as the skill of the interviewer and their ability to build trust for productive communication. Our choice of paragraph representation, the initial gut reaction question, and the interview question regarding whether participants felt their mental model had changed were all designed to help assess whether we were achieving our target and to provide more data about early changes in the data mental model.

Through our semi-structured interview discussions, we found a mix of participants, some who claimed their mental model was unchanged and others where the changes were apparent from their words.

Some participants expressed confusion in response to the initial question after they read the dataset: "What was your gut reaction or intuition about the dataset?" While we intended to capture open-ended responses, participants sometimes asked us to clarify what we meant by "gut reaction." This confusion may have influenced their responses or the responses of others who were confused but did not vocalize their confusion.

5.5.3 Implications for the Data Visualization Design Process

We share implications about mental models and participants' thoughts on data for the visualization community.

Personal Mental Models

Beware of data abstraction elephants. We observed diverse data types and visual representations arising from our study datasets (code C1). Even in the file system case, where most participants cited a similar source of inspiration, participants sketched a variety of concepts and imparted differing grouping biases. Across all datasets, some participants were influenced by recent events in their life (e.g., labmate's talk) or by expectations of what 'data' in general should be, rather than the dataset at hand (code F3, code D2). These realities present hazards in choosing effective data abstractions. We suggest that designers sample multiple target users, potentially multiple times, so that our interpretation of our users' mental models can solidify. Once solid, we can better identify what data abstractions best align with these mental models.

Visualizing and discussing help elicit a person's mental model comprehensively. The varying levels of verbal descriptions of their sketches (code E8), the assortment of terminology used about their data abstractions (code E7), and the range of data abstractions (code C1) suggest that people can generally explain their mental model well but need multiple avenues to externalize it. The discussion with Participant 014 (see code E7) and prior work with visualizing genome sequences [108] show that observing problem-solving can expose underlying aspects of the mental model.

We observed that participants tended to overestimate what is "obvious" in their mental model, a psychological phenomenon shown in a visualization setting by Xiong et al. via a controlled study [159]. The lack of legends (code E5) suggests that the sketch is truly the user's mental model of the dataset, but makes understanding the sketches difficult for anyone but the sketch's author. We suggest visualization designers solicit conversations and sketches about the dataset—*not* of chart or representation types—from their users. Centering the conversation on the dataset, rather than a representation, will focus the conversation.

Different abstractions support different mental affordances, indicating tasks. Although visualization researchers tend to consider interactivity in the context of a visual design, people readily described their mental models in interactive terms, often with only loose—if any—association to specific visual encodings (code C7).

We suggest designers consider the interplay between the tasks that different data types readily afford and note the interactions that the users describe. For example, a table lends itself naturally to sorting, whereas a graph lends itself more naturally to navigation. The affordances of the specific data abstraction that a person latches onto may betray the tasks they most need to perform. Conversely, a given data abstraction may inspire specific, predictable forms of purpose-seeking (code F5). Ensuring task and data abstractions are aligned may translate to more intuitive interactions and more effective visualization designs.

People's views about what data *is* and what it *isn't* may limit ideas during data creation. Several participants related data to tables, computers, Excel, or database tables (code D1). They had definitions for what "data" is and what it isn't (code D2). This may limit or expand the data abstraction. These ideas are important to discuss during data reconnaissance [34] and throughout the creation of the data abstraction. We suggest visualization designers provoke users by proposing alternative data abstractions, sources, and formats that may help expand the definition of data and uncover latent data abstractions [13].

Before Designing Visualizations

Visualization design starts with data design. Many of our participants imagined beyond the dataset. They suggested possible sources of the data (code F6), invented tasks to be done with the data (code F5), and wished for additional data or information (code F4). This curiosity may be due to the participants having no relationship to the dataset, but could also be due to inherent curiosity.

This creativity can be useful to visualization designers, as it highlights what aspects of the data the user finds relevant and the tasks and operations the user wants to execute. If possible, when encountering a situation with “no real data available (yet)” [133], visualization designers have a chance to be part of the data design phase. We define the *data design phase* as an unconventional part of the Design phase in the design study methodology where the contents, attributes, and format of the data is still under discussion even though the design study process has begun. In such a scenario, visualization designers should engage in these conversations with their collaborators. By being present in these discussions, designers can better understand the priorities and motivations of their users. An alternative would be to have the users recreate the dataset from memory; the features and entries that the user finds most important will likely be remembered.

Extra care must be taken when eliciting data abstractions with relations between data items. We observed participants used a variety of terms and visual representations when there were relations between data items (code C6). Some used terms that were inconsistent with the visualization community’s definitions (code E7). It was difficult to confidently determine the participant’s mental model despite their language and sketch. Even when there were similar visual representations among sketches, such as in the file system dataset, the ways they were spoken about were different.

These observations suggest that visualization designers should practice extra care when eliciting data abstraction when relations are present. One example may not be enough to determine the nature of the data described. If we were to probe further, a set of relation assertions (e.g., “connections like this may never occur”) may elicit more detail. However, there are some abstractions where the structures may be the same but the meaning and conventions are different, such as trees and hierarchies, where this approach alone would not be enough.

The way people express relations between data items suggests a continuum of data abstractions. Visualization designers may be able to leverage

this fluidity. Another way to interpret our observations regarding how people organize relations between data items (code C6) is that the data abstraction classifications represent examples in a continuum of data abstractions. This continuum does not seem to have clearly defined axes but instead seems to be a continuous space of how they organize data without the strong boundaries associated with data type classifications and taxonomies. Datasets do not need to fit one named abstraction. We observed Participant 030 refine their data abstraction from a hierarchy to a network when considering how the dataset might be stored in a database. Participant 018 spoke of sets and hierarchies together. This suggests there may be utility in representations that allow people to leverage these multiple ways of abstracting the relationship structures in the data.

Suggesting multiple audiences can elicit multiple data abstractions. When the topic of communicating with other people was discussed we observed participants changing their sketches, or claiming they would, sometimes to the point of selecting a different data abstraction. This behavior could be employed to explore several useful abstractions of the same data. There may be differences among what they would sketch for themselves, how they would communicate *their mental model* to someone else, and how they would communicate the data to someone else—or potentially multiple such "someone elses."

Multiple audiences may also help identify cases where people bow to expectations about how they are "supposed to" visualize data. We observed expectations regarding what is considered "data," including tables, demographics, and counts. Such conventions may lead to less useful abstractions. Prior work in creative visualization workshops and collaborative prototyping provides a framework for facilitating exploring alternative, useful design ideas [78, 39, 40].

5.6 Conclusion

We presented a sketch-based study to probe people's mental models and their corresponding data abstractions before they are given visual or other structural cues. We observed diversity and fluidity in the mental models that participants described and the data abstractions and visual representations that the participants used. This diversity can be influenced by several factors including examples from their lives, common approaches to the context, things they had seen recently, imagined tasks, and their conceptions of what "data" is and the conventions that come with it. We also observed that participants used a variety of terms and relations to describe the data and their sketches and would reconfigure their model when considering different audiences. These observations suggest that care must be taken when eliciting descriptions of data for the process of data abstraction and visualization design, but also offer options for leveraging the data design process to further probe user needs and possible abstractions, as well as opportunities to use the framework of communication to explore the data exploration space.

Family and Friends Summary: Our work in the previous section shows that the choice of data abstraction can change how the user understands the visualization. The choice of data abstraction is important – a visualization designer could base their design on a tabular abstraction, but the user might know of hidden connections in the dataset and so may think of the data as a connected network. If so, the table-based visualization that the designer made will be more difficult for the user to interact with and understand. This multi-perspective take on datasets is similar to the fable about the elephant. Five people in the dark encounter an elephant for the first time, and they each come to a different conclusion (e.g., the one who touches the tusks thinks an elephant is like a spear).

We conducted a study to see what conclusions people would make from our "data elephant", i.e., our cleverly designed datasets in *paragraph form.* After reading the dataset, the participant sketched their mental model of the data (to see the datasets and try your own sketch, jump to subsection 5.3.3). We saw all sorts of data representations and data abstractions appear in these sketches, even though the participants used the same dataset. The variety and creativity shown by the participants can help visualization designers better understand how their users think about the data, which results in a more useful visualization for the user.

Theme	Code	Representative example/Evidence
Depictions of mental models	Used text to clarify (E3)	Participant 009 (FS) calls the files ".java" so that "we can be more explicit."
	Legends/verbal legends (E5)	Participant 006 (FS) verbally explained the icons used for file types. Participant 032 (PS) made a legend.
	Used abstractions in depiction (sometimes laziness, sometimes deliberate) (E1)	Participant 008 (PS): "...just repeating the picture of the power plant but then I got lazy and then just drew a square for the power plants."
	Constrained by sketches (E2)	Participant 007 (JD): added Sharpies outside of pencil pouch because "I didn't make the pouch big enough." Participant 016 (JD): "I had to skip back here [to the left] to fill out the rest of the space."
	Added details to clarify (E4)	Participant 020 (PS): suggests adding weights to directed arrows, "it would be an improvement."
Communication with others	Conflict of the terminology used (E7)	Participant 006 (FS) drew something closer to a node-link diagram but their description was more set-like ("levels", "within", "in the folder").
	Range of description detail from literal to abstract/overview (E8)	5 participants did not describe the drawing and instead restated the dataset (4/5 were CS-related participants). 15 participants gave a visual description of the icons and marks (13/15 were CS-related participants). 8 participants named a data abstraction or data chart type (4/8 were CS-related participants).
	Changes to depiction for communication (E6)	Participant 021 (FS) said they would draw the tree left-to-right to better communicate with others and attempted to add interaction indications.

Table 5.3: *Themes about Mental Model Elicitation.* This table contains our themes and codes about mental model elicitation, as well as representative examples for each code. The codes are labeled as "E#". The complete list of codes and all supporting evidence can be found at `https://osf.io/kvnb9/` and in the supplemental material.

Theme	Code	Representative example/Evidence
Mental model formation process	Immediate mental model formation (F1)	Participant 024 (PS): in response to the first question, "My gut reaction was like an image of—I dunno if you know cell-free MIMO graphs..."
	Mental model did not vary much or at all for 17/28 participants (F1) * Caveat: changed significantly for 4 participants due to changes in data abstraction representation (F1)	Participant 018 (FS): "No, I mean I kind of saw it for how I was gonna do it right away and stuck with that." Participant 016 (JD) considered "maybe there's a better way to accurately display—because it sounds like this is someone inventorying the items... I wish there was a way that I could highlight that, or draw attention to that this [the stamp roll] is probably more important than rubber bands and tacks."
	Mental models became more detailed (F2)	Participant 013 (JD): "In the beginning, I was just thinking about the basket and then I started to remember how things were more clearly, so I started drawing slightly more elaborately and really thinking about what I wanted to draw."
Influences on the mental model formation	Explicit mental model origins (F3)	Participant 006 (FS): "The [Windows] file system, the file structure, has definitely left a mark on me."
	Purpose-seeking: Participants added or assumed tasks (F5)	Participant 019 (JD) associated the dataset with cleaning or organizing their desk. Participants 011, 014, 032 (PS) all sought to discover ultima (e.g., the maximum power produced).
	Purpose-seeking: Participants suggested data sources (F6)	Participant 017: "I dunno, it's a power station, it's probably a municipal guide or a power company's guide to how to distribute power."
	Purpose-seeking: Participants desired to add data/information (F4)	Add data attribute: Participant 029 (PS) wanted to add people per house.
		Add relationships: Participant 013 (JD) wanted to add relationships between items.
		Add naming schema: Participant 018 (FS) wanted to add folder names.
	Lower math literacy works against the mental model (F7)	Participant 023 had difficulty with multidimensional aspect of power station data, dropped the power stations' label dimension (i.e.,A–F labels).

Table 5.4: Themes about Mental Model Formation. This table contains our themes and codes about mental model formation, as well as some representative examples for each code. The codes are labeled as "F#". The complete list of codes and all supporting evidence can be found at `https://osf.io/kvnb9/` and in the supplemental material.

CHAPTER 6

Conclusion

In this dissertation, we explored different levels for understanding data: internal structures (i.e., mental models), abstract scaffolding (i.e., data abstractions), external visuals (i.e., data visualizations). Internal mental models contain personal connections to the data, incorporating the user's knowledge on the topic, prior experience with similar data, and personal interests in the data. Data abstractions provide a way to connect these specific mental models to more general structures, like a tabular, network, or spatial data abstraction. Data abstractions are established concepts in the data and visualization worlds, so there exists terminology to describe these concepts and default common visualizations used to display these abstractions. Visualization designers use data abstractions to "translate" personal mental models into more common abstractions that both the user and the designer can understand.

We investigated the flexibility and relationships between mental models, data abstractions, and visualizations throughout this dissertation to explore how the visualization community can best design for our users. We observed this malleability in three scenarios. First, in the design of the tree visualization Atria in chapter 3, our users' consistent mental model of a tree allowed us keep the tree visualization that we had developed. We preserved the extra graph edges to accurately represent the data, but our users' mental models and understanding of relationships in programs meant that we could show these edges on demand, instead of permanently displaying them. This case shows how visualization designers can leverage their user's mental model to guide design decisions.

Next, during our survey of data abstractions in chapter 4, we saw that domain experts were able to shift their dataset from one data abstraction to another, although they did not always enjoy the process. Changing abstractions often required adding properties to the data, like adding a timestamp or creating edges between data. While this transition required extra work and creative thinking, it allowed for new insights to arise from the data. Sensitivity and care should be taken by the visualization designer when undertaking a similar exercise because our domain experts often had a preferred data abstraction and an established data pipeline, leading to resistance when considering alternatives.

Finally, in our study of mental models in chapter 5, we saw participants voluntarily include different data abstractions and visualization representations when discussing different use cases of the data. Participants considered how an alternative audience might view the visualization and described how they would make modifications. Our participants added qualities to the data and invented tasks for the data, which unveiled a hidden creativity and curiosity about the data. Data design is not usually emphasized in the design study methodology as a task for both the visualization designer and the domain expert; however, our results show that exploring the data "jungle" together with the domain expert can elicit useful information about the data and influence visualization design choices.

While we did not define a set method for eliciting mental models in the visualization design methodology, we gained insight into the mental models of domain experts and novice users and connected these mental models to more concrete data abstractions. Our users view our visualizations and understand data with a variety of backgrounds, knowledge about types of visualizations, and ways of interacting with data. If we can better understand how the user understands, problem-solves, and manipulates their data, we can design visualizations that more closely align with their existing mental processes and workflow.

6.1 Limitations and Future Work

This dissertation provides initial work in how the mental model of a dataset can be used to influence visualization design, benefiting both our users and ourselves as visualization designers. Each of these three works supports the idea of probing our users about their mental models of data using external artifacts (e.g., a prototype of a graph, a list of possible data abstractions, a sketch). In these situations, we deliberately challenged our users' assumptions of the data and visualization by provoking responses like "That wouldn't work because...", "That makes no sense", or "Obviously I drew it this way...". In this work, we advocate for sincerely and creatively coming together with our users to uncover their insights and opinions and about their data, *before* any visualization influences discussions. Our results have shown how this lateral approach to data design and visualization design can reveal data insights and data abstraction preferences that might not have come up in conversation.

Changing our approach to selecting data abstractions based on the person-centric mental model versus the data-centric dataset can potentially improve the clarity of our designs, leading to our users more easily adopting the visualization. This approach can also help us as designers to better understand how our users think about and interact with the data, leading to a more streamlined design process and a stronger understanding of the expected tasks. Throughout our interactions with data workers, we observed that suggesting a concrete abstraction, particularly one that was unlike how the data worker usually conceptualized their data, elicited rich feedback about their data and their thinking on it.

In chapter 3, we showed a case study in adjusting the design in accordance to the mental model. This detailed example showed the steps we took to succeed in a pitfall-laden environment. We provide guidelines for other practitioners to follow but recognize the uniqueness of our situation. When appropriate, we encourage fostering a synergistic environment between visualization designers and their collaborators to that traditionally challenging visualization projects can succeed in spite of the pitfalls.

In chapter 4, our choice of data abstractions for our data typology was both a weakness and a strength. Our participants in the survey left feedback when they felt they needed to use a different abstraction or if they thought their data did not fit any one particular abstraction. Ultimately, these comments and discussions in interviews would not have happened if we did not present a typology in the first place. Further work is needed to evaluate how to effectively harness exploring the abstraction choice with one's collaborators during the design phase.

In chapter 5, during each sketching interview, we presented the participant with a small dataset in paragraph form. These datasets may be an unrealistic representation of real-world data for analysis, but given that many visualization designers are given small "toy" datasets at the start, we felt our datasets presented a similar scenario to our participants. Exploring larger, more complicated datasets would be an interesting future direction of this work to see how the participant aggregated the data or divided it into sub-units. More work is needed to test various methods and questions for eliciting mental models in formal design study scenarios and to develop guidelines as to how to conduct these discussions.

Family and Friends Summary: We explored the jungle of data abstractions with different perspectives. With the tree visualization Atria in chapter 3, our users' mental model helped us move from a graph abstraction to a tree abstraction. This showed us data abstractions can morph, which is a useful technique for visualization designers and an idea that we further tested through the survey in chapter 4. We prompted data experts to think of their dataset using an alternative data abstraction. The data experts were able to morph their dataset into a different data abstraction, finding new angles to look at their dataset, even though the change might not be part of their workflow. Finally, we conducted a study where we presented a paragraph of data for the participant to sketch, to see what data abstractions and representations they used. Each participant had a unique perspective on the dataset, resulting in different drawings, inventing a purpose for the dataset, and creating tasks to perform on the data (e.g., find the power station that produces the most power). Visualization designers can use their user's knowledge and creativity before the design process to better understand the scope of the dataset and the tasks the user wants to accomplish.

This dissertation opens up ideas and discussions for visualizations designers and users about data so that the user ultimately receives the best visualization for their needs. Through the prior chapters, we demonstrated evidence that proposing alternative data abstractions can spark useful conversations about data and tasks, and that users can help in this process by providing their expertise and creativity. This brainstorming about *data*, rather than about visualization designs, should happen early in the design process as it gives both the user and the designer a better understanding of the dataset and the priorities and tasks the user has for the dataset. Ultimately, these discussions could help streamline later aspects of the design process and result in visualizations that better fit the users mental model. The next step for this line of thought would be to include data abstraction and data discussions during the design study process, and to reflect and evaluate the functionality of the discussions.

CHAPTER 7

List of Published Works

Submitted papers under review (i.e. not published) are denoted with *.

1. K. Williams, “Different Data Elephants,” presented at Data Blitz session at Women in Data Science Tucson Virtual Conference, April 22, 2022.

2. K. Williams, A. Bigelow and K. E. Isaacs. 2023. “Data Abstraction Elephants: The Initial Diversity of Data Idioms and Mental Models,” To appear in Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI ’23).

3. * C. Scully-Allison, I. Lumsden, K. Williams, J. Bartels, M. Taufer, S. Brink, A. Bhatele, O. Pearce, and K. E. Isaacs, “Designing an Interactive, Notebook-Embedded Tree Visualization to Support Exploratory Performance Analysis,” submitted to VIS 2023.

4. A. Bigelow, K. Williams, and K. E. Isaacs, “Guidelines For Pursuing and Revealing Data Abstractions,” IEEE Transactions on Visualization and Computer Graphics, vol. 27, no. 2, pp. 1503-1513, 2021. doi: 10.1109/TVCG.2020.3030355

5. S. Brink, I. Lumsden, C. Scully-Allison, K. Williams, O. Pearce, T. Gamblin, M. Taufer, K. Isaacs, and A. Bhatele, “Usability and Performance Improvements in Hatchet,” presented at the ProTools 2020 Workshop, held in conjunction with the International Conference for High Performance Computing, Networking, Storage and Analysis (SC ‘20), November 2020, held virtually.

6. S. Brandt, A. Bigelow, S. Sakin, K. Williams, K. E. Isaacs, K. Huck, R. Tohid, B. Wagle, S. Shirzad and H. Kaiser, “JetLag: An Interactive, Asynchronous Array Computing Environment,” To appear in Proceedings of the Practice and Experience on Advanced Research Computing (PEARC ’20). Association for Computing Machinery, New York, NY, USA, 2020.

7. K. Williams, A. Bigelow and K. E. Isaacs, “Visualizing a Moving Target: A Design Study on Task Parallel Programs in the Presence of Evolving Data and Concerns,” IEEE Transactions on Visualization and Computer Graphics, vol. 26, no. 1, pp. 1118-1128, 2020. doi: 10.1109/TVCG.2019.2934285

8. R. Tohid, B. Wagle, S. Shirzad, P. Diehl, A. Serio, A. Kheirkhahan, P. Amini, K. Williams, K. E. Isaacs, K. Huck, S. Brandt, H. Kaiser, “Asynchronous Execution of Python Code on Task Based Runtime Systems,” IEEE/ACM 4th International Workshop on Extreme Scale Programming Models and Middleware (ESPM2’18), SC ’18, 2018, pp. 37-45, doi: 10.1109/ESPM2.2018.00009.

REFERENCES

[1] A. Adamoli and M. Hauswirth. Trevis: A context tree visualization & analysis framework and its use for classifying performance failure reports. In *Proceedings of the 5th International Symposium on Software Visualization*, SoftVis, pp. 73–82. ACM, New York, NY, USA, 2010. doi: 10.1145/1879211.1879224

[2] D. H. Ahn, B. R. de Supinski, I. Laguna, G. L. Lee, B. Liblit, B. P. Miller, and M. Schulz. Scalable temporal order analysis for large scale debugging. In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, SC, pp. 44:1–44:11. ACM, New York, NY, USA, 2009. doi: 10.1145/1654059.1654104

[3] J. Annett, K. D. Duncan, R. B. Stammers, and M. J. Gray. Task analysis. Training Information Number 6. London: HMSO, 1971.

[4] J. Annett and N. A. Stanton. *Research and developments in task analysis*, pp. 1–8. CRC Press, 2000.

[5] L. Bartram, M. Correll, and M. Tory. Untidy Data: The Unreasonable Effectiveness of Tables. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):686–696, 2022.

[6] M. Bauer, S. Treichler, E. Slaughter, and A. Aiken. Legion: Expressing Locality and Independence with Logical Regions. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC '12, pp. 66:1–66:11. IEEE Computer Society Press, 2012.

[7] A. Bergel, A. Bhatele, D. Boehme, P. Gralka, K. Griffin, M.-A. Hermanns, D. Okanović, O. Pearce, and T. Vierjahn. Visual analytics challenges in analyzing calling context trees. In A. Bhatele, D. Boehme, J. A. Levine, A. D. Malony, and M. Schulz, eds., *Programming and Performance Visualization Tools*, pp. 233–249. Springer International Publishing, Cham, 2019.

[8] A. Bigelow. *Driving Genetics with Experimental Visualization.* Undergraduate thesis, University of Utah, Salt Lake City, UT, USA, 2012.

[9] A. Bigelow. Reflections on Working With Fellow Tool Builders. In *Proceedings of the IEEE Information Visualization Conference - Posters (InfoVis '17)*, 2017.

[10] A. Bigelow, S. Drucker, D. Fisher, and M. Meyer. Reflections on how designers design with data. In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces*, AVI '14, pp. 17–24. ACM, 2014. doi: 10.1145/2598153.2598175

[11] A. Bigelow, S. Drucker, D. Fisher, and M. Meyer. Reflections on how designers design with data. In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces*, AVI '14, pp. 17–24. ACM, 2014. doi: 10.1145/2598153.2598175

[12] A. Bigelow, C. Nobre, M. D. Meyer, and A. Lex. Origraph: Interactive network wrangling. *IEEE VAST*, 2019.

[13] A. Bigelow, K. Williams, and K. E. Isaacs. Guidelines for Pursuing and Revealing Data Abstractions. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1503–1513, Feb. 2021. doi: 10.1109/TVCG.2020.3030355

[14] J. E. Block and E. D. Ragan. Micro-entries: Encouraging deeper evaluation of mental models over time for interactive data systems. In *2020 IEEE Workshop on Evaluation and Beyond - Methodological Approaches to Visualization (BELIV)*, pp. 38–47. IEEE Computer Society, Los Alamitos, CA, USA, oct 2020. doi: 10.1109/BELIV51497.2020.00012

[15] G. Bosilca, A. Bouteiller, A. Danalis, M. Faverge, T. Herault, and J. J. Dongarra. PaRSEC: Exploiting Heterogeneity to Enhance Scalability. *Computing in Science and Engg.*, 15(6):36–45, Nov. 2013. doi: 10.1109/MCSE.2013.98

[16] M. Bostock, V. Ogievetsky, and J. Heer. D3: Data-Driven Documents. *IEEE Trans. on Vis. and Comp. Graphics*, 17(12):2301–2309, Dec. 2011. doi: 10.1109/TVCG.2011.185

[17] G. E. Box. Robustness in the strategy of scientific model building. In *Robustness in Statistics*, pp. 201–236. Academic Press, 1979. `https://linkinghub.elsevier.com/retrieve/pii/B9780124381506500182`. doi: 10.1016/B978-0-12-438150-6.50018-2

[18] M. Brehmer and T. Munzner. A multi-level typology of abstract visualization tasks. *IEEE Trans. Visualization and Computer Graphics (TVCG) (Proc. InfoVis)*, 19(12):2376–2385, 2013.

[19] S. Brink, I. Lumsden, C. Scully-Allison, K. Williams, O. Pearce, T. Gamblin, M. Taufer, K. E. Isaacs, and A. Bhatele. Usability and performance improvements in hatchet. In *2020 IEEE/ACM International Workshop on HPC User Support Tools (HUST) and Workshop on Programming and Performance Visualization Tools (ProTools)*, pp. 49–58, 2020. doi: 10.1109/HUSTProtools51951.2020.00013

[20] S. Brinkmann, J. Gracia, and C. Niethammer. Task Debugging with Temanejo. In *Tools for High Performance Computing 2012*, pp. 13–21. Springer, 2013.

[21] K. R. Butcher. Learning from text with diagrams: Promoting mental model development and inference generation. *Journal of Educational Psychology*, 98(1):182–197, 2006. doi: 10.1037/0022-0663.98.1.182

[22] F. Campello, B. Pinto, G. Tessarolli, A. Oliveira, C. Oliveira, M. O. Junior, L. Murta, and V. Braganholo. A similarity-based approach to match elements across versions of XML documents. *Brazilian Symposium on Databases (SBBD)*, p. 10, 2014.

[23] M. Carcary. The Research Audit Trial–Enhancing Trustworthiness in Qualitative Inquiry. *Electronic Journal of Business Research Methods*, 7(1), 2009.

[24] M. Carcary. The research audit trial–enhancing trustworthiness in qualitative inquiry. *Electronic Journal of Business Research Methods*, 7(1), 2009.

[25] S. K. Card and D. Nation. Degree-of-interest trees: A component of an attention-reactive user interface. In *AVI'02: Proceedings of the Working Conference on Advanced Visual Interfaces*, pp. 231–245. ACM Press, 2002. doi: 10.1145/1556262.1556300

[26] S. Chandrasegaran, D. Ramanujan, and N. Elmqvist. How do sketching and non-sketching actions convey design intent? In *Proceedings of the 2018 Designing Interactive Systems Conference*, DIS '18, pp. 373–385. ACM, New York, NY, USA, 2018. doi: 10.1145/3196709.3196723

[27] K. Charmaz. *Constructing Grounded Theory*. sage, 2014.

[28] E. Chi. A taxonomy of visualization techniques using the data state reference model. In *IEEE Symposium on Information Visualization 2000. INFOVIS 2000. Proceedings*, pp. 69–75, Oct. 2000. doi: 10.1109/INFVIS.2000.885092

[29] M. Chin. FILE NOT FOUND: A generation that grew up with Google is forcing professors to rethink their lesson plans. The Verge, Sept 2021.

[30] N. J. Cooke. Varieties of knowledge elicitation techniques. *International Journal of Human-Computer Studies*, 41(6):801–849, 1994. doi: 10.1006/ijhc.1994.1083

[31] N. J. Cooke and A. L. Rowe. Evaluating mental model elicitation methods. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 38(4):261–265, 1994. doi: 10.1177/154193129403800416

[32] I. Corporation. Open Community Runtime. https://01.org/open-community-runtime, October Access March 2019.

[33] A. Crisan, J. L. Gardy, and T. Munzner. On regulatory and organizational constraints in visualization design and evaluation. In *Proc. ACM BELIV Workshop*, pp. 1–9, 2016.

[34] A. Crisan and T. Munzner. Uncovering data landscapes through data reconnaissance and task wrangling. In *2019 IEEE Visualization Conference (VIS)*, pp. 46–50, 2019. doi: 10.1109/VISUAL.2019.8933542

[35] M. Da Gandra and M. Van Neck. *InformForm: Information Design: In Theory, an Informed Practice*. Mwmcreative Limited, July 2012.

[36] J. de St. Germain, J. McCorquodale, S. Parker, and C. Johnson. Uintah: A Massively Parallel Problem Solving Environment. In *Ninth IEEE International Symposium on High Performance and Distributed Computing*, pp. 33–41. IEEE, Piscataway, NJ, Nov. 2000.

[37] S. Devkota and K. E. Isaacs. CFGExplorer: Designing a Visual Control Flow Analytics System around Basic Program Analysis Operations. *Computer Graphics Forum (Proceedings of EuroVis 2018)*, 37(3), 2018. doi: 10.1111/cgf.13433

[38] J. Dokulil and J. Katreniakova. Visualization of Open Community Runtime Task Graphs. In *Proceedings of the 2017 21st International Conference on Information Visualisation*, IV, 2017.

[39] S. Dow, J. Fortuna, D. Schwartz, B. Altringer, D. Schwartz, and S. Klemmer. Prototyping dynamics: Sharing multiple designs improves exploration, group rapport, and results. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pp. 2807–2816. ACM, New York, NY, USA, 2011. doi: 10.1145/1978942.1979359

[40] S. P. Dow, A. Glassco, J. Kass, M. Schwarz, D. L. Schwartz, and S. R. Klemmer. Parallel prototyping leads to better design results, more divergence, and increased self-efficacy. *ACM Trans. Comput.-Hum. Interact.*, 17(4):1–24, dec 2011. doi: 10.1145/1879831.1879836

[41] P. Eklund, N. Roberts, and S. Green. OntoRama: Browsing RDF ontologies using a hyperbolic-style browser. In *First International Symposium on Cyber Worlds, 2002. Proceedings.*, pp. 405–411, Nov. 2002. doi: 10.1109/CW.2002.1180907

[42] M. Feinberg. A design perspective on data. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pp. 2952–2963. Association for Computing Machinery, Denver, Colorado, USA, May

2017. https://doi.org/10.1145/3025453.3025837. doi: 10.1145/3025453.3025837

[43] A. Figueiras. A typology for data visualization on the web. In *2013 17th International Conference on Information Visualisation*, pp. 351–358, July 2013. doi: 10.1109/IV.2013.45

[44] A. Fox, E. Vries, L. Lima, and S. Loker. Exploring representations of student time-use. In *Diagrammatic Representation and Inference - 9th International Conference, Diagrams 2016, Philadelphia, PA, USA, August 7-10, 2016, Proceedings*, pp. 40–47, 08 2016. doi: 10.1007/978-3-319-42333-3_4

[45] T. Gamblin. "Personal OCD: when people use "tree" to refer to a graph that isn't necessarily a tree. Your "dependency tree" is usually a directed acyclic graph (DAG). Do you do this? Why/why not?". Twitter, September 2021.

[46] E. R. Gansner, E. Koutsofios, S. C. North, and K.-p. Vo. A Technique for Drawing Directed Graphs. *IEEE Trans. on Soft. Eng.*, 19(3):214–230, 1993.

[47] E. R. Gansner and S. C. North. An open graph visualization system and its applications to software engineering. *Software – Prac. and Exp.*, 30(11):1203–1233, 2000.

[48] J. G. Geer. Do open-ended questions measure "salient" issues? *Public Opinion Quarterly*, 55(3):360–370, Jan. 1991. https://academic.oup.com/poq/article/55/3/360/1927045. doi: 10.1086/269268

[49] K. I. Gero, Z. Ashktorab, C. Dugan, Q. Pan, J. Johnson, W. Geyer, M. Ruiz, S. Miller, D. R. Millen, M. Campbell, S. Kumaravel, and W. Zhang. Mental models of ai agents in a cooperative game setting. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, p. 1?12. Association for Computing Machinery, New York, NY, USA, 2020. doi: 10.1145/3313831.3376316

[50] M. Ghoniem, J.-D. Fekete, and P. Castagliola. A comparison of the readability of graphs using node-link and matrix-based representations. In *Proc. IEEE Symp. on Info. Vis.*, pp. 17–24, 2004.

[51] S. Goodwin, J. Dykes, S. Jones, I. Dillingham, G. Dove, A. Duffy, A. Kachkaev, A. Slingsby, and J. Wood. Creative user-centered visualization design for energy analysts and modelers. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2516–2525, Dec. 2013. doi: 10.1109/TVCG.2013.145

[52] T. Götschi, I. Sanders, and V. Galpin. Mental models of recursion. *SIGCSE Bull.*, 35(1):346 – 350, jan 2003. doi: 10.1145/792548.612004

[53] P. J. Guo, S. Kandel, J. M. Hellerstein, and J. Heer. Proactive wrangling: Mixed-initiative end-user programming of data transformation scripts. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pp. 65–74. Association for Computing Machinery, Santa Barbara, California, USA, Oct. 2011. `https://doi.org/10.1145/2047196.2047205`. doi: 10.1145/2047196.2047205

[54] M. C. Hao, M. Hsu, U. Dayal, and A. Krug. Web-Based Visualization of Large Hierarchical Graphs Using Invisible Links in a Hyperbolic Space. In H. Arisawa and T. Catarci, eds., *Advances in Visual Information Management: Visual Database Systems. IFIP TC2 WG2.6 Fifth Working Conference on Visual Database Systems May 10–12, 2000, Fukuoka, Japan*, pp. 83–94. Springer US, Boston, MA, 2000. doi: 10.1007/978-0-387-35504-7_6

[55] S. Harper and S. Dorton. Towards a context-dependent framework for visualizing mental models. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 64(1):308–312, 2020. doi: 10.1177/1071181320641071

[56] B. Haugen, S. Richmond, J. Kurzak, C. Steed, and J. Dongarra. Visualizing Execution Traces with Task Dependencies. In *Proceedings of the 2nd Workshop on Visual Performance Analysis*, VPA, Nov. 2015.

[57] J. Heer and A. Perer. Orion: A system for modeling, transformation and visualization of multidimensional heterogeneous networks. In *IEEE Visual Analytics Science \& Technology (VAST)*, p. 10, 2011.

[58] M. Hegarty and M. A. Just. Constructing mental models of machines from text and diagrams. *Journal of memory and language*, 32(6):717–742, 1993.

[59] U. Hinrichs, S. Forlini, and B. Moynihan. Speculative Practices: Utilizing InfoVis to Explore Untapped Literary Collections. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):429–438, Jan. 2016. doi: 10.1109/TVCG.2015.2467452

[60] S. Huron, S. Carpendale, J. Boy, and J.-D. Fekete. Using VisKit: A Manual for Running a Constructive Visualization Workshop. In *Pedagogy of Data Visualization Workshop at IEEE VIS 2016*. Baltimore, MD, United States, Oct. 2016.

[61] S. Huron, Y. Jansen, and S. Carpendale. Constructing visual representations: Investigating the use of tangible tokens. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2102–2111, 2014. doi: 10.1109/TVCG.2014.2346292

[62] E. L. Hutchins, J. D. Hollan, and D. A. Norman. Direct manipulation interfaces. *Human-Computer Interaction*, 1(4):311–338, 1986.

[63] H. Hutchinson, W. Mackay, B. Westerlund, B. B. Bederson, A. Druin, C. Plaisant, M. Beaudouin-Lafon, S. Conversy, H. Evans, H. Hansen, N. Roussel, and B. Eiderbäck. Technology Probes: Inspiring Design for and with Families. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pp. 17–24. ACM, 2003. doi: 10.1145/642611.642616

[64] A. Huynh, D. Thain, M. Pericas, and K. Taura. DAGViz: A DAG Visualization Tool for Analyzing Task-Parallel Program Traces. In *Proceedings of the 2nd Workshop on Visual Performance Analysis*, VPA, Nov. 2015.

[65] K. E. Isaacs, A. Bhatele, J. Lifflander, D. Böhme, T. Gamblin, M. Schulz, B. Hamann, and P.-T. Bremer. Recovering Logical Structure from Charm++ Event Traces. In *Proceedings of the ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '15, Nov. 2015. LLNL-CONF-670046.

[66] K. E. Isaacs and T. Gamblin. Preserving Command Line Workflow for a Package Management System using ASCII DAG Visualization. *To appear in IEEE Transactions on Visualization and Computer Graphics*, 2019. doi: 10.1109/TVCG.2018.2859974

[67] K. E. Isaacs, A. Giménez, I. Jusufi, T. Gamblin, A. Bhatele, M. Schulz, B. Hamann, and P.-T. Bremer. State of the Art of Performance Visualization. In *Eurographics/IEEE Conference on Visualization State-of-the-Art Reports*, EuroVis '14, 2014.

[68] T. J. Jankun-Kelly and K.-L. Ma. MoireGraphs: Radial Focus+Context Visualization and Interaction for Graphs with Visual Nodes. In *Proceedings of the Ninth Annual IEEE Conference on Information Visualization*, INFOVIS'03, pp. 59–66. IEEE Computer Society, 2003.

[69] P. N. Johnson-Laird. Mental models in cognitive science. *Cognitive Science*, 4(1):71–115, 1980. doi: 10.1016/S0364-0213(81)80005-5

[70] D. H. Jonassen and P. Henning. Mental models: Knowledge in the head and knowledge in the world. *Educational Technology*, 39:37–42, 1999.

[71] N. A. Jones, H. Ross, T. Lynam, P. Perez, and A. Leitch. Mental models: an interdisciplinary synthesis of theory and methods. *Ecology and Society*, 16:46–46, 2011.

[72] H. Kaiser, B. Adelstein-Lelbach, et al. HPX source code repository, 2007. Available under the Boost Software License (a BSD-style open source license).

[73] H. Kaiser, T. Heller, B. Adelstein-Lelbach, A. Serio, and D. Fey. HPX: A Task Based Programming Model in a Global Address Space. In *Proceedings of the 8th International Conference on Partitioned Global Address Space Programming Models*, PGAS '14, pp. 6:1–6:11. ACM, 2014. doi: 10.1145/2676870.2676883

[74] L. V. Kale and S. Krishnan. CHARM++: A Portable Concurrent Object Oriented System Based on C++. In *Proceedings of the Eighth Annual Conference on Object-Oriented Programming Systems, Languages, and Applications*, OOPSLA '93, pp. 91–108. ACM, 1993. doi: 10.1145/165854.165874

[75] S. Kandel, A. Paepcke, J. Hellerstein, and J. Heer. Wrangler: Interactive visual specification of data transformation scripts. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 3363–3372, 2011. doi: 10.1145/1978942.1979444

[76] R. Keller, C. M. Eckert, and P. J. Clarkson. Matrices or node-link diagrams: Which visual representation is better for visualizing connectivity models? *Information Visualization*, 5:62–76, 2006.

[77] E. Kerzner, S. Goodwin, J. Dykes, S. Jones, and M. Meyer. A framework for creative visualization-opportunities workshops. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):748–758, Jan. 2019. doi: 10.1109/TVCG.2018.2865241

[78] E. Kerzner, S. Goodwin, J. Dykes, S. Jones, and M. Meyer. A framework for creative visualization-opportunities workshops. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):748–758, 2019. doi: 10.1109/TVCG.2018.2865241

[79] D. Kirsh. *Thinking with External Representations*, pp. 61–84. Springer, 03 2017. doi: 10.1007/978-3-319-49115-8_4

[80] Á. Kiss and T. Szirányi. Evaluation of manually created ground truth for multi-view people localization. In *Proceedings of the International Workshop on Video and Image Ground Truth in Computer Vision Applications*, VIGTA '13, pp. 1–6. Association for Computing Machinery, St. Petersburg, Russia, July 2013. `https://doi.org/10.1145/2501105.2501106`. doi: 10.1145/2501105.2501106

[81] G. Klein and R. Hoffman. Macrocognition, mental models, and cognitive task analysis methodology. In J. M. Schraagen, L. G. Militello, T. Ormerod, and

R. Lipshitz, eds., *Naturalistic Decision Making and Macrocognition*, chap. 4, pp. 57–80. CRC Press, Boca Raton, 01 2008.

[82] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, and C. Willing. Jupyter Notebooks – a publishing format for reproducible computational workflows. In F. Loizides and B. Schmidt, eds., *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, pp. 87–90. IOS Press, 2016.

[83] A. Knüpfer, C. Rössel, D. an Mey, S. Biersdorff, K. Diethelm, D. Eschweiler, M. Geimer, M. Gerndt, D. Lorenz, A. Malony, et al. Score-P: A joint performance measurement run-time infrastructure for Periscope, Scalasca, TAU, and Vampir. In *Tools for High Performance Computing 2011*, pp. 79–91. Springer, 2012.

[84] H. Lam, M. Tory, and T. Munzner. Bridging from goals to tasks with design study analysis reports. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):435–445, 2018. doi: 10.1109/TVCG.2017.2744319

[85] J. Lamping, R. Rao, and P. Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In I. R. Katz, R. Mack, L. Marks, M. B. Rosson, and J. Nielsen, eds., *CHI'95: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 401–408. ACM Press/Addison-Wesley Publishing Co., 1995. doi: 10.1145/223904.223956

[86] B. Lee, K. Isaacs, D. A. Szafir, G. E. Marai, C. Turkay, M. Tory, S. Carpendale, and A. Endert. Broadening intellectual diversity in visualization research papers. *IEEE Computer Graphics and Applications*, 39(4):78–85, July 2019. doi: 10.1109/MCG.2019.2914844

[87] B. Lee, C. Plaisant, C. S. Parr, J.-D. Fekete, and N. Henry. Task Taxonomy for Graph Visualization. In *Proceedings of the 2006 AVI BELIV Workshop*, BELIV '06, pp. 1–5. ACM, 2006. doi: 10.1145/1168149.1168168

[88] C. Lewis. A model of mental model construction. *SIGCHI Bull.*, 17(4):306–313, apr 1986. doi: 10.1145/22339.22388

[89] S. Lin, F. Taïani, T. C. Ormerod, and L. J. Ball. Towards anomaly comprehension: Using structural compression to navigate profiling call-trees. In *Proceedings of the 5th International Symposium on Software Visualization*, SOFTVIS, pp. 103–112. ACM, New York, NY, USA, 2010. doi: 10.1145/1879211.1879228

[90] Z. Liu, S. B. Navathe, and J. T. Stasko. Ploceus: Modeling, visualizing, and analyzing tabular data as networks. *Information Visualization*, 13(1):59–89, Jan. 2014. doi: 10.1177/1473871613488591

[91] Z. Liu and J. T. Stasko. Mental Models, Visual Reasoning and Interaction in Information Visualization: A Top-down Perspective. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):999–1008, Nov. 2010. doi: 10.1109/TVCG.2010.177

[92] E. Mayr, G. Schreder, M. Smuc, and F. Windhager. Looking at the representations in our mind: Measuring mental models of information visualizations. In *Proceedings of the Sixth Workshop on Beyond Time and Errors on Novel Evaluation Methods for Visualization*, BELIV '16, pp. 96–103. Association for Computing Machinery, New York, NY, USA, 2016. doi: 10.1145/2993901.2993914

[93] N. McCurdy, J. Dykes, and M. Meyer. Action design research and visualization design. In *Proceedings of the Sixth Workshop on beyond Time and Errors on Novel Evaluation Methods for Visualization*, BELIV '16, pp. 10–18. ACM, 2016. doi: 10.1145/2993901.2993916

[94] S. McKenna. *The Design Activity Framework: Investigating the Data Visualization Design Process*. PhD thesis, University of Utah, June 2017.

[95] S. McKenna, D. Mazur, J. Agutter, and M. Meyer. Design activity framework for visualization design. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2191–2200, Dec. 2014. doi: 10.1109/TVCG.2014.2346331

[96] M. Meyer and J. Dykes. Reflection on Reflection in Applied Visualization Research. *IEEE Computer Graphics and Applications*, 38(6):9–16, 2018. doi: 10.1109/MCG.2018.2874523

[97] M. Meyer and J. Dykes. Criteria for rigor in visualization design study. *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–1, 2019. doi: 10.1109/TVCG.2019.2934539

[98] M. Meyer, M. Sedlmair, P. S. Quinan, and T. Munzner. The nested blocks and guidelines model. *Information Visualization*, 14(3):234–249, July 2015. `http://journals.sagepub.com/doi/10.1177/1473871613510429`. doi: 10.1177/1473871613510429

[99] S. Milgram and D. Jodelet. Psychological maps of paris: (1970). In *The People, Place, and Space Reader*, pp. 45–49. Routledge, Abingdon-on, 2014.

[100] A. Moravcsik. Active citation: A precondition for replicable qualitative research. *PS: Political Science & Politics*, 43(1):29–35, Jan. 2010. doi: 10.1017/S1049096510990781

[101] M. Muller. *Curiosity, Creativity, and Surprise as Analytic Tools: Grounded Theory Method*, pp. 25–48. Springer New York, New York, NY, 2014. doi: 10.1007/978-1-4939-0378-8_2

[102] M. Muller, I. Lange, D. Wang, D. Piorkowski, J. Tsay, Q. V. Liao, C. Dugan, and T. Erickson. How data science workers work with data: Discovery, capture, curation, design, creation. In *Proceedings of the 2019 CHI conference on human factors in computing systems*, pp. 1–15, 2019.

[103] T. Munzner. A nested model for visualization design and validation. *IEEE Transactions on Visualization and Computer Graphics*, 15(1):921–928, 2009.

[104] T. Munzner. *Visualization Analysis and Design.* CRC Press, Boca Raton, FL, 12 2014. doi: 10.1201/b17511

[105] T. Munzner, A. Endert, A. Lex, A. Ynnerman, C. Garth, M. Chen, P. Isenberg, and L. Shixia. Revise committee town hall. `https://drive.google.com/drive/u/0/folders/1dqssldHbXLmAD9ze0qHCbfNTb8gjeHKS`, Oct. 2019.

[106] T. Munzner, F. Guimbretiére, S. Tasiran, L. Zhang, and Y. Zhou. TreeJuxtaposer: Scalable tree comparison using Focus+Context with guaranteed visibility. *ACM Transactions on Graphics*, 22(3):453–462, 2003. doi: 10.1145/882262.882291

[107] H. T. Nguyen, L. Wei, A. Bhatele, T. Gamblin, D. Boehme, M. Schulz, K.-L. Ma, and P.-T. Bremer. VIPACT: a visualization interface for analyzing calling context trees. In *Proceedings of the Third International Workshop on Visual Performance Analysis*, VPA, pp. 25–28, 2016. doi: 10.1109/VPA.2016.9

[108] C. B. Nielsen, S. D. Jackman, I. Birol, and S. J. Jones. ABySS-Explorer: Visualizing Genome Sequence Assemblies. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):881–888, 2009. doi: 10.1109/TVCG.2009.116

[109] C. Nobre, N. Gehlenborg, H. Coon, and A. Lex. Lineage: Visualizing Multivariate Clinical Data in Genealogy Graphs. *IEEE Transactions on Visualization and Computer Graphics*, 25(3):1543–1558, 2019. doi: 10.1109/TVCG.2018.2811488

[110] C. Nobre, M. Streit, and A. Lex. Juniper: A Tree+Table Approach to Multivariate Graph Visualization. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '18)*, 25(1):544–554, Jan. 2019. doi: 10.1109/TVCG.2018.2865149

[111] C. Nobre, M. Streit, M. Meyer, and A. Lex. The state of the art in visualizing multivariate networks. *Computer Graphics Forum (EuroVis)*, 38:807–832, 2019. doi: 10.1111/cgf.13728

[112] S. Passi and S. Jackson. Data vision: Learning to see through algorithmic abstraction. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing - CSCW '17*, pp. 2436–2447. ACM Press, Portland, Oregon, USA, 2017. `http://dl.acm.org/citation.cfm?doid=2998181.2998331`. doi: 10.1145/2998181.2998331

[113] S. Passi and S. J. Jackson. Trust in data science: Collaboration, translation, and accountability in corporate data science projects. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW):136:1–136:28, Nov. 2018. `http://doi.org/10.1145/3274405`. doi: 10.1145/3274405

[114] E. M. Peck, S. E. Ayuso, and O. El-Etr. Data is personal: Attitudes and perceptions of data visualization in rural pennsylvania. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, pp. 1–12. Association for Computing Machinery, Glasgow, Scotland Uk, May 2019. `https://doi.org/10.1145/3290605.3300474`. doi: 10.1145/3290605.3300474

[115] K. H. Pine and M. Liboiron. The politics of measurement and action. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pp. 3147–3156. Association for Computing Machinery, Seoul, Republic of Korea, Apr. 2015. `https://doi.org/10.1145/2702123.2702298`. doi: 10.1145/2702123.2702298

[116] V. G. Pinto, L. Stanisic, A. Legrand, L. M. Schnorr, S. Thibault, and V. Danjean. Analyzing Dynamic Task-Based Applications on Hybrid Platforms: An Agile Scripting Approach. In *Proceedings of the Third International Workshop on Visual Performance Analysis*, VPA, 2016.

[117] C. Plaisant, J. Grosjean, and B. B. Bederson. SpaceTree: Supporting Exploration in Large Node Link Tree, Design Evolution and Empirical Evaluation. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02)*, INFOVIS '02, pp. 57–. IEEE Computer Society, 2002.

[118] E. S. Poole, M. Chetty, R. E. Grinter, and W. K. Edwards. More than meets the eye: Transforming the user experience of home network management. In *Proceedings of the 7th ACM Conference on Designing Interactive Systems*, DIS '08, p. 455?464. Association for Computing Machinery, New York, NY, USA, 2008. doi: 10.1145/1394445.1394494

[119] A. J. Pretorius and J. J. Van Wijk. What does the user want to see? what do the data want to be? *Information Visualization*, 8(3):153–166, Sept. 2009. `https://doi.org/10.1057/ivs.2009.13`. doi: 10.1057/ivs.2009.13

[120] E. M. Reingold and J. S. Tilford. Tidier Drawings of Trees. *IEEE Trans. Softw. Eng.*, 7(2):223–228, Mar. 1981. doi: 10.1109/TSE.1981.234519

[121] N. Reissmann, M. Jahre, and A. Muddukrishna. Towards Aggregated Grain Graphs. In *Proceedings of the Fourth International Workshop on Visual Performance Analysis*, VPA, Nov. 2017.

[122] U. Reja, K. L. Manfreda, V. Hlebec, and V. Vehovar. Open-ended vs. close-ended questions in web questionnaires. *Developments in Applied Statistics*, 19(1):159–177, 2003.

[123] A. Rind, W. Aigner, M. Wagner, S. Miksch, and T. Lammarsch. Task cube: A three-dimensional conceptual space of user tasks in visualization design and evaluation. *Information Visualization*, 15(4):288–300, 2016. doi: 10.1177/1473871615621602

[124] J. C. Roberts, C. Headleand, and P. D. Ritsos. Sketching designs using the five design-sheet methodology. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):419–428, 2016. doi: 10.1109/TVCG.2015.2467271

[125] S. A. Sakin, A. Bigelow, R. Tohid, C. Scully-Allison, C. Scheidegger, S. R. Brandt, C. Taylor, K. A. Huck, H. Kaiser, and K. E. Isaacs. Traveler: Navigating task parallel traces for performance analysis. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):788–797, 2023. doi: 10.1109/TVCG.2022.3209375

[126] P. Salmon, D. Jenkins, N. Stanton, and G. Walker. Hierarchical task analysis vs. cognitive work analysis: comparison of theory, methodology and contribution to system design. *Theoretical Issues in Ergonomics Science*, 11(6):504–531, 2010. doi: 10.1080/14639220903165169

[127] M. Sandelowski. Real qualitative researchers do not count: The use of numbers in qualitative research. *Research in Nursing & Health*, 24(3):230–240, June 2001. doi: 10.1002/nur.1025

[128] I. Sanders, V. Galpin, and T. Götschi. Mental models of recursion revisited. *SIGCSE Bull.*, 38(3):138 –142, jun 2006. doi: 10.1145/1140123.1140162

[129] W. Schnotz and M. Bannert. Construction and interference in learning from multiple representation. *Learning and Instruction*, 13(2):141–156, 2003. External and Internal Representations in Multimedia Learning. doi: 10.1016/S0959-4752(02)00017-8

[130] T. L. Scholtz and I. Sanders. Mental models of recursion: Investigating students' understanding of recursion. In *Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education*, ITiCSE '10, p. 103–107. Association for Computing Machinery, New York, NY, USA, 2010. doi: 10.1145/1822090.1822120

[131] G. Schreder, F. Windhager, M. Smuc, and E. Mayr. Supporting cognition in the face of political data and discourse: A mental models perspective on designing information visualization systems. In *2016 Conference for E-Democracy and Open Government (CeDEM)*, pp. 213 – 218. IEEE Computer Society, Krems, Austria, 2016. doi: 10.1109/CeDEM.2016.23

[132] H. Schulz. Treevis.net: A Tree Visualization Reference. *IEEE Computer Graphics and Applications*, 31(6):11–15, Nov. 2011. doi: 10.1109/MCG.2011.103

[133] M. Sedlmair, M. Meyer, and T. Munzner. Design study methodology: Reflections from the trenches and the stacks. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2431–2440, 2012. doi: 10.1109/TVCG.2012.213

[134] S. Shende and A. Malony. The TAU Parallel Performance System. *IJHPCA*, 20(2, Summer):287–311, 2006. ACTS Collection Special Issue.

[135] A. Srinivasan, H. Park, A. Endert, and R. C. Basole. Graphiti: Interactive specification of attribute-based edges for network modeling and visualization. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):226–235, Jan. 2018. doi: 10.1109/TVCG.2017.2744843

[136] R. J. Sternberg. *Handbook of Creativity.* Cambridge University Press, 1999.

[137] A. Sukumaran and C. Nass. Socially cued mental models. In *CHI '10 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '10, pp. 3379 – 3384. Association for Computing Machinery, New York, NY, USA, 2010. doi: 10.1145/1753846.1753988

[138] A. Thudt, U. Hinrichs, S. Huron, and S. Carpendale. Self-reflection and personal physicalization construction. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pp. 1–13. Association for Computing Machinery, New York, NY, USA, 2018. doi: 10.1145/3173574.3173728

[139] Tittha sutta, Udāna 6.4, Khuddaka Nikaya. E-book, Oct 2022.

[140] R. Tohid, B. Wagle, S. Shirzad, P. Diehl, A. Serio, A. Kheirkhahan, P. Amini, K. Williams, K. Isaacs, K. Huck, S. Brandt, and H. Kaiser. Asynchronous Execution of Python Code on Task Based Runtime Systems. In *Proceedings of the Fourth International IEEE Workshop on Extreme Scale Programming Models and Middleware*, Nov. 2018.

[141] C. Tominski, J. Abello, F. van Ham, and H. Schumann. Fisheye Tree Views and Lenses for Graph Visualization. In E. Banissi, R. A. Burkhard, A. Ursyn, J. J. Zhang, M. Bannatyne, C. Maple, A. J. Cowell, G. Y. Tian, and M. Hou, eds., *IV'06: Proceedings of the International Conference on Information Visualisation*, pp. 17–24. IEEE Computer Society, 2006. doi: 10.1109/IV.2006.54

[142] J. G. Trafton, S. B. Trickett, and F. E. Mintz. Connecting internal and external representations: Spatial transformations of scientific visualizations. *Foundations of Science*, 10(1):89–106, 2005. doi: 10.1007/s10699-005-3007-4

[143] B. Tversky. Visualizing thought. In *Handbook of human centric visualization*, pp. 3–40. Springer, 2014.

[144] Unknown. Tittha sutta, Udāna 6.4, Khuddaka Nikaya. E-book, Oct 2022.

[145] R. Verborgh and M. D. Wilde. *Using OpenRefine*. Packt Publishing Ltd, Sept. 2013.

[146] R. Von Oech and G. Willett. *A Kick in the Seat of the Pants: Using Your Explorer, Artist, Judge, & Warrior to Be More Creative*. Perennial Library, 1986.

[147] B. Wagle, M. A. H. Monil, K. Huck, A. D. Malony, A. Serio, and H. Kaiser. Runtime Adaptive Task Inlining on Asynchronous Multitasking Runtime Systems. *48th International Conference on Parallel Processing (ICPP 2019)*, Aug. 2019. doi: 10.1145/3337821.3337915

[148] J. Walny, S. Carpendale, N. H. Riche, G. Venolia, and P. Fawcett. Visual Thinking In Action: Visualizations As Used On Whiteboards. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2508–2517, Dec. 2011. doi: 10.1109/TVCG.2011.251

[149] J. Walny, C. Frisson, M. West, D. Kosminsky, S. Knudsen, M. S. T. Carpendale, and W. Willett. Data changes everything: Challenges and opportunities in data visualization design handoff. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):12–22, 2020. doi: 10.1109/TVCG.2019.2934538

[150] J. Walny, C. Frisson, M. West, D. Kosminsky, S. Knudsen, S. Carpendale, and W. Willett. Data changes everything: Challenges and opportunities in data visualization design handoff. *arXiv:1908.00192 [cs]*, July 2019. `http://arxiv.org/abs/1908.00192`.

[151] J. Walny, S. Huron, and S. Carpendale. An Exploratory Study of Data Sketching for Visual Representation. *Computer Graphics Forum*, 34(3):231–240, june 2015. doi: 10.1111/cgf.12635

[152] C. Ware. *Information Visualization: Perception for Design.* Morgan Kaufmann Publishers Inc., San Francisco, 2nd ed., 2004. doi: 10.1016/B978-155860819-1/50001-7

[153] M. Weed. Capturing the essence of grounded theory: The importance of understanding commonalities and variants. *Qualitative Research in Sport, Exercise and Health*, 9(1):149–156, Jan. 2017. doi: 10.1080/2159676X.2016.1251701

[154] C. Wiener. *Making Teams Work in Conducting Grounded Theory*, pp. 292–310. SAGE Publications LTD, London, England, 2007. doi: 10.4135/9781848607941

[155] K. Williams, A. Bigelow, and K. Isaacs. Data abstraction elephants. In *Proceedings of the 2023 CHI conference on human factors in computing systems*, pp. 1–22, 2023.

[156] K. Williams, A. Bigelow, and K. E. Isaacs. Visualizing a moving target: A design study on task parallel programs in the presence of evolving data and concerns. *To appear in IEEE Transactions on Visualization and Computer Graphics (Proceedings of InfoVis '19)*, Jan. 2020. doi: 10.1109/TVCG.2019.2934285

[157] J. Wood, R. Beecham, and J. Dykes. Moving beyond Sequential Design: Reflections on a Rich Multi-Channel Approach to Data Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2171–2180, Dec. 2014. doi: 10.1109/TVCG.2014.2346323

[158] T. Wun, J. Payne, S. Huron, and S. Carpendale. Comparing bar chart authoring with microsoft excel and tangible tiles. *Computer Graphics Forum*, 35:111–120, 06 2016. doi: 10.1111/cgf.12887

[159] C. Xiong, L. Van Weelden, and S. Franconeri. The curse of knowledge in visual data communication. *IEEE Transactions on Visualization and Computer Graphics*, 26(10):3051–3062, 2020. doi: 10.1109/TVCG.2019.2917689

[160] C. Yehezkel, M. Ben-Ari, and T. Dreyfus. Computer architecture and mental models. *SIGCSE Bull.*, 37(1):101 – 105, feb 2005. doi: 10.1145/1047124.1047390

[161] A. X. Zhang, M. Muller, and D. Wang. How do data science workers collaborate? roles, workflows, and tools. *To appear in ACM Human-Computer Interaction*, 4(CSCW), May 2020. http://arxiv.org/abs/2001.06684.

[162] Y. Zhang, K. Chanana, and C. Dunne. Idmvis: Temporal event sequence visualization for type 1 diabetes treatment decision support. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):512–522, Jan 2019. doi: 10.1109/TVCG.2018.2865076

[163] C. Ziemkiewicz and R. Kosara. The shaping of information by visual metaphors. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1269–1276, 2008. doi: 10.1109/TVCG.2008.171