VisPrac: Provenance for Collaborative Data Exploration

Kaustubh Odak[*]
Pune Institute of Computer Technology

Tanusri Bhowmick[†]
Pune Institute of Computer Technology

Yash Sonar[‡]
Pune Institute of Computer Technology

Hussain Burhanuddin[§]
Pune Institute of Computer Technology

Alark Joshi[¶]
University of San Francisco

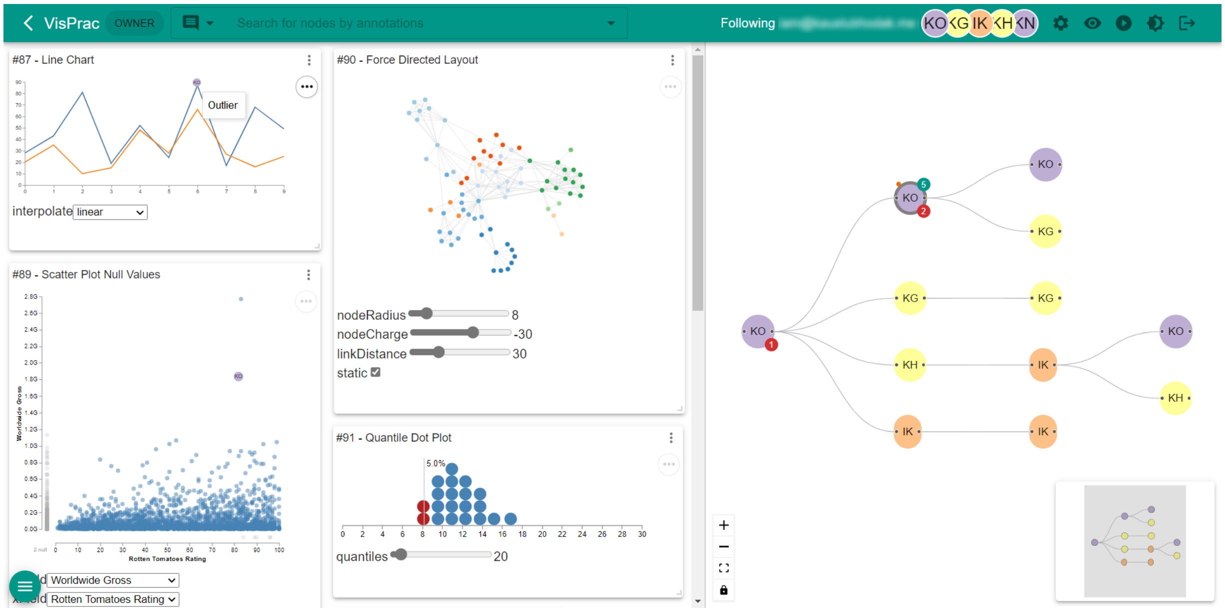


Figure 1: This figure shows the VisPrac interface for collaborative exploration that shows active users, annotations, bookmarks, multiple resizable and draggable visualizations, a provenance tree and a minimap for the same. The navigation bar has information about the current user's role in the project (Owner) and the user being 'followed' by them (Following *username*).

Abstract

Visualizations are often used by a team in the process of information foraging. Collaborators are often unaware of the intermediate findings of the other members and the steps that they took to reach those findings. We present, VisPrac, a collaborative interface to explore visualizations with a provenance tree to track the changes made by each user. Our system allows users to explore and annotate various visualizations, annotate states of the provenance tree to provide a rationale, and follow individual users as they analyze the data. The provenance tree enables participants to visit previous states of the exploration process and provides accountability in the decision-making process.

Index Terms: Human-centered computing—Visualization systems and tools; Human-centered computing—Visual Analytics

1 Introduction

Data-driven decision making relies on team-based collaborative exploration of data using visualizations. In some cases, the steps taken to reach a decision are not available to all the members of the team. Provenance is defined as the history of ownership of actions or events [9]. Provenance has been used in the visualization field to keep a record of the actions of an individual as they explore data [7, 15, 16, 20].

Pirolli and Card [18] introduced the *sensemaking loop* where an analyst transitions from information foraging to hypothesis generation and, eventually, insight. Ericsson and Lehmann [5] state that "Experts don't just automatically extract patterns .. from memory. Instead, they select the relevant information and *encode* it... that allows planning, evaluation, and reasoning". While the analytic process can lead to an analyst immersing themselves in the information foraging process, the ability to review the process and gain a better understanding of it for planning and evaluation is crucial for accountability and reproducibility of the foraging process.

To facilitate collaborative data exploration and sensemaking, we developed *VisPrac*. Visprac allows multiple users to collaboratively explore data while being able to track the changes made by each user in the form of a provenance tree. VisPrac allows individuals to bookmark nodes in the provenance tree, add annotations to the visualizations and the nodes in the provenance tree, follow a user as they interact with the data, expand/collapse subtrees for large provenance trees, highlight all the nodes created by a user, invite other participants, review a user's path to a node using the slideshow mode, search for nodes that contain specific annotations, and explore data in incognito mode for hypothesis testing.

[*] e-mail: kaustubhodak1@gmail.com
[†] e-mail: tanusribhowmick0@gmail.com
[‡] e-mail: yashsonar213@gmail.com
[§] e-mail: hussainburhanuddin1@gmail.com
[¶] e-mail: apjoshi@usfca.edu

2 Related Work

2.1 Analytic Provenance

Heer et al. [7] introduced the concept of Graphical Histories to capture the interaction and data analysis history. It can be be used to communicate findings and to gain a deeper understanding of the analytical process. North et al. [16] introduced *analytic provenance* as an area of research to understand the reasoning process of analysts. Herschel [8] discussed the various types of provenance and found that it increases *understandability* among collaborators, increases *recall* and *reproducibility*, and lastly, improves the *quality* due to improved assessment and increased trust in the process. Nguyen et al. [15] introduced SensePath - an analytic provenance system that aims to automatically analyze the sensemaking process through the various views included in the system. Their team later introduced SenseMap [14] that captured visualization and user actions as well as allowed participants to create their own knowledge map. While participants found SenseMap intuitive to use, it did not have any collaborative features.

Ragan and Goodall [20] stated that provenance provides a way to store the process memory and communication among collaborators. Ragan et al. [21] found that providing visual history aids (even low resolution views) through provenance leads to analysts remembering their process for explanation to colleagues. In a subsequent paper, Ragan et al. [19] proposed a framework that classifies provenance by types and purpose. Among the various types of provenance mentioned in the framework, *visualization* and *interaction* provenance is captured in VisPrac with potential for *rationale* and *insights* through the annotations feature. Madanagopal et al. [10] interviewed analysts and identified provenance as one of the key components for effective collaboration.

Trrack [3] is a library for tracking provenance in web-based applications. Trrack stores nodes sequentially in an object, and references to child nodes in an array. Differential states are used to track changes between nodes. VisPrac tracks provenance in the form of a tree that grows as multiple collaborators explore data independently or collaboratively.

2.2 Annotations

Annotations have been found to be extremely useful in the field of collaborative visual analytics [11, 12]. Missier et al. [12] found three different types of annotations - Precision, Focus, and Optimization to capture findings when exploring lineaage. InsideInsights [11] is a system that provides a hierarchical insight management system that facilitates annotation of states, structure, and links between them to presentation views. Zhao et al. [23] explored the challenges with *handoff* in asynchronous collaboration. Annotations and playback of history were found to be useful for the knowledge transfer. Playback of history is available in VisPrac through the Slideshow Mode feature. Chung et al. [1] introduced Vizcept to facilitate synchronous collaborative exploration and visualization of large text datasets. While Vizcept contains facilities for collaborative exploration, there is no provenance and no ability to explore non-textual data.

2.3 Collaborative Sensemaking

Park et al. [17] posit that displaying the analysis process along with the visual presentations enables users to reduce ambiguity and keeps a record of the decisions made along the exploratory process. Coelho et al. [2] introduced the share.va framework to store and share of the states of visual analytics dashboards through blockchain, to ensure security and trust among collaborators.

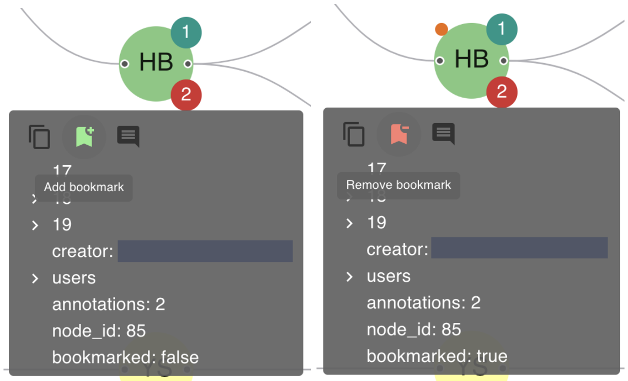
3 Approach

VisPrac enables synchronous collaboration for data exploration and decision-making. It requires participants to sign up or login in order to save states under their account. Participants can either create a new project or join an existing project for which they have received an invitation by email.

While exploring the visualizations the users can annotate their findings, add other visualizations or choose to follow another user. Any changes to the visualizations are noted and saved in the provenance tree created on the right. Figure 1 shows the interface with the visualizations on the left and the provenance tree on the right. The tree stores each state as a node with the creator as the owner of the root node. A state is defined as a collection of all the visualizations and the current settings of the parameters in those visualization. The node is a Vega [13, 22] state stored on the back end along with the creator ID, the parent node ID, the visualization chart number it belongs to, the timestamp of its creation, and the timestamp of its update. On the front end, each node represents the current state of each visualization, the creator of the node, the people examining the same state, and the annotation count. The edge labels between the nodes encode the changes made to reach that node from its parent node. For example, if a slider was moved for a visualization from 0.5 to 1.3, the edge label displays the visualization ID that contains the slider and the new value of the slider such as **19 - step: 1.3**. To minimize clutter, the edge labels are only visible on hover or when a user zooms into the provenance tree.

3.1 Collaboration

The creator of the project is the owner and can set the project's access rights as public or private. If the project access is set to private, only invited members can access the project. Collaborators can be invited to the project by email. The URL of the project can also be shared by copying it. A user can be a collaborator or a viewer. If the user is a collaborator then the user can interact with the visualizations, thus impacting the provenance tree. If the user is a viewer then the user can only view the visualizations and the provenance tree as others interact with it, but they cannot make any changes to the visualizations.



17
› 18
› 19
creator: [redacted]
› users
annotations: 2
node_id: 85
bookmarked: false

17
› 18
› 19
creator: [redacted]
› users
annotations: 2
node_id: 85
bookmarked: true

Figure 2: Node Design: The left figure shows the 'Add Bookmark' icon button that can be used to bookmark nodes. It also shows that the node has one user examining the state (green circle with a 1 on the top right) and two annotations (red circle with a 2 on the bottom right). The right figure shows a bookmarked node. The orange circle in the top-left of the node is the bookmark indicator. The 'Remove Bookmark' button is also shown in the right figure.

3.2 Creating and Storing the State

The collaborative interface is created using Vega specifications that are processed to extract key-value pairs with the key 'signals'. From these, only the signals with which the user can interact are stored based on their 'on' or 'bind' property in its definition object. These signals are sequentially passed to Vega's View API, which returns the initial value of each signal. All signal key-value pairs are grouped according to their corresponding visualizations, and stored in a nested object. This object is then used as the initial state of the root node of the project.

Except the initial state of the root node, which contains the default values of all the meaningful signals in the project, each state stores a delta. In VisPrac, a delta is a simple JSON object which stores the differences between two states in the provenance tree. This delta is computed using the jsondiffpatch [4] library.

When a project is opened, the deltas of all the existing nodes are patched sequentially on the states of their parent nodes, starting from the root node. In essence, the entire provenance tree is reconstructed. This is done only once when the user opens the project to reduce the run-time complexity of patching nodes in real-time. When a new node is added to the provenance tree, the delta of this new node is patched on the state of its parent node on other users' machines. This enables quick switching between nodes to examine their respective states.

3.3 Provenance Tree

Nodes are added into the provenance tree as users interact with the visualizations. The interaction is with the signals provided in the Vega specification. Signals such as cursor, events, and input binding [6] lead to the addition of new nodes into the provenance tree. Hovering over a node in the provenance tree shows a dialog box that shows (a) A list of all the visualizations (identified by a unique ID), (b) The number of users currently on the same node, (c) The annotations in that state, and (d) The ID of the node. Figure 2 shows the result of the hover action on a node.

When reviewing the provenance tree, subtrees can be collapsed by right-clicking on a node to manage large subtrees that may not be of interest.

3.3.1 Provenance Tree Node Design

A node in the provenance tree contains encodings to convey information about the state. The right image in Figure 2 shows a node that has one user currently examining the state (as shown by the green circle with a 1 in it), the state contains two annotations (as shown by the red circle with a 2 in it), and that the node is bookmarked (as shown by the orange dot on the top left of the node.)

Bookmarks provide an important way for analysts to curate and capture their findings during the sensemaking process. Creating bookmarks in Visprac is straightforward and requires the user to hover the cursor over the node that is to be bookmarked and to click the green 'Add bookmark' icon button in the tooltip. Figure 2 shows a screenshot of the add/remove bookmarks feature. Bookmarks can be removed by clicking the red 'Remove bookmark' icon button in the tooltip.

3.4 Database decisions

Initially, our project used Trrack [3], a provenance-tracking state-management library. Trrack's support for Firebase-driven persistence encouraged us to use the Firebase Realtime Database, and later, Cloud Firestore for data persistence and synchronization. The unstructured/NoSQL data model allowed us to directly store projects and provenance trees in the form of documents and collections. Using their realtime API, we were able to synchronize the users' and the project's statuses across multiple clients. As the scope of our project increased, we started noticing instability in our implementation. Trrack's approach of storing bi-directional references of provenance tree nodes and importing/exporting the entire tree was making it difficult for us to make/retrieve consistent and granular updates to/from the database. An increase in the number of nodes and users would eventually cause Trrack to fluctuate between intermediate states of the collaborative interface.

To resolve this issue, we migrated to Supabase, a PostgreSQL-based Backend-as-a-Service. Overhauling our state-management to work with an SQL-driven solution allowed us to leverage its ACID capabilities. We were able to make data synchronization more granular and specific by using PostgreSQL's logical replication functionality with normalized tables. We also created a basic form of IAM (Identity and Access Management) using PostgreSQL's row-level security policies.

Interactions performed by each user on the charts reflect as a new node on the provenance tree, provided there exists a signal for the specific interaction in the respective chart. Every chart might have a set of Vega signals which when triggered are captured by the platform. Every captured signal along with the user ID is recorded and sent to Supabase. Supabase stores the interactions in a PostgreSQL database which gets reflected to all the other users. These stored interactions show up as new nodes on the provenance tree. Each node contains information of the state of the entire collaborative interface at that instant and the user can click on the node to move to that particular state. Simultaneous changes to the same root node by different users result in different nodes and a user can create multiple branches from the same root node.

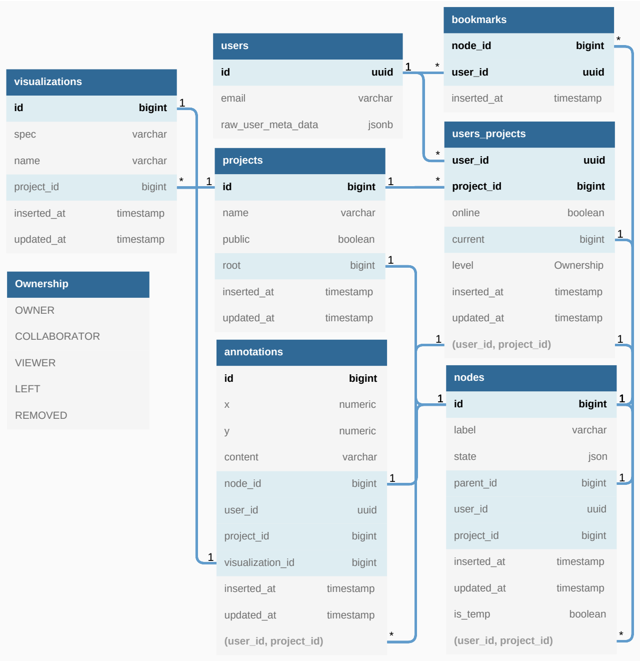


Figure 3: This figure shows the database schema employed by VisPrac.

Figure 3 shows our database schema. Our implementation contains the following tables: (a) User information and metadata is stored in the 'users' table. (b) The 'project' table stores the name, visibility (public/private) and the root node of the project. (c) The names and specifications of the Vega visualizations in the project are stored in the 'visualization' table. (d) The many-to-many relationship between users and projects is facilitated via a junction table ('users_projects'). When a project is created, the creator is added to the users_projects table as the owner, and a root node, with signals from all the visualizations under the project, is created. (e) The ownership of a node is determined by referencing the composite primary key of users_projects as a foreign key in the nodes table. (f) The structure of the provenance tree is stored using a self-referential/recursive relationship between the parent and children nodes. (g) The 'annotations' table stores the coordinates and the content of the annotation, as well as the associated node's, creator's, project's and visualization's IDs. (h) The 'bookmarks' table identifies the nodes that users have bookmarked in different projects using the nodes' and users' IDs.

3.5 Multiple users

For synchronous collaboration between multiple users, we need the updates to reach all collaborators. This requires propagating updates and receiving updates. The system follows client-server architecture, where each collaborator is a client to a database server. The collaboration system must be real-time and support multiple collaborators working at the same time, to achieve this low latency and concurrency control is a must. The database server is a SQL database providing concurrency control and consistency among clients. Updates made are new nodes of the provenance tree, which are added to the database, available for the collaborators.

Receiving changes must also be in real-time and Collaborators need to update their provenance trees based on updates made. To achieve this, each collaborator sets-up listeners for updates, instead of querying every few seconds. On receiving changes, collaborators' views update their provenance tree based on the current state of the database.

3.6 Annotations

A Collaborator can add an annotation to any visualization or node in the provenance tree. To add an annotation to a visualization, they need to open the menu of the target visualization and click on 'Add annotation.' This is followed by adding the text for the annotation, clicking on the desired location on the visualization for the annotation, and confirming the action. Figure 4 shows the process of adding an annotation to a Quantile Dot Plot.

To implement annotations, custom data, marks and signals are injected in the Vega specifications at run-time. The annotation is constructed using an `svg` group of symbols and text, defined by data and marks in Vega. To draw the viewers' attention to the annotations, a pulsating effect is added using CSS filters and transforms. The location of an annotation is determined using custom Vega signals, which capture the coordinates of user-driven mouse events. On creation, annotations are coupled with the active node and the target visualization using foreign keys to their corresponding tables. This helps in determining the right place and time for displaying annotations.

Users can also search for nodes using the annotation content as search query. When the query is entered in the search field, a list of matching nodes are displayed right below it. To jump to the node with the desired annotation content, users can simply select that node from the list of results. That node is then selected as the current node, and also centered in the provenance tree.

3.7 Finding node through annotations

Users can search for all the nodes that contain a user-specified annotation. The search feature is found on the top left of the interface and can be see on the top left in Figure 1. The node which the annotation belongs to is centered on the screen to reduce the need for the users to look for that node in a large provenance tree. Upon clicking the node, the annotations are made visible and the user can view the corresponding visualizations in that state.

3.8 Slideshow Mode

When collaborating on a project, the members of that project can verify the work of their collaborators for accountability. To review the steps taken to reach a specific node, we added the Slideshow mode. The user clicks on a node of their choice and then clicks the "Slideshow" play button on the navigation bar. The slideshow begins from the root node and gradually traverses the tree towards the selected node. As the slideshow progresses the visualizations show the state corresponding to each node as the route is being traversed.

3.9 Data Exploration in Incognito Mode

We facilitate the ability for an individual to test hypotheses through the "Incognito Mode" feature. When a user enters "Incognito Mode," (shown in Figure 5) the interactions of that user are not available to the rest of the collaborators, but they are visible to the user and are stored in the database with a flag indicating that the node is temporary. Only the user who created the node (while being in incognito mode) will be able to see the temporary nodes connected by dotted edges, as shown in Figure 5.

After the user is done exploring the data in incognito mode, they can exit the incognito mode. At this point, the user is presented with a dialog box asking the user "Do you want to save your progress?". If the user selects "Yes," then all the new nodes created in that incognito session are added to the provenance tree and the the flag indicating temporary presence in the database is reset. This results in those nodes to be visible to everyone in the project. If the user selects "No," then all the new nodes for that incognito session labeled with the temporary flag are deleted from the database and the provenance tree remains unchanged.

4 Conclusion & Future Work

We present the VisPrac system that allows collaborative exploration of data with provenance for accountability and reproducibility. Users can annotated findings and rationale for their decisions and novices can follow experts in the system as they analyze data. The provenance tree keeps a detailed history of the state of the visualizations and the interactions that led to the various states of the collaborative interface.

In the future, we plan to evaluate the benefits of a provenance tree as compared to a linear layout that contains a list of all the changes made by the collaborators in the project.

References

[1] H. Chung, S. Yang, N. Massjouni, C. Andrews, R. Kanna, and C. North. Vizcept: Supporting synchronous collaboration for constructing visualizations in intelligence analysis. In *2010 IEEE Symposium on Visual Analytics Science and Technology*, pp. 107–114. IEEE, 2010.

[2] D. Coelho, R. Trailor, D. Sill, S. Engle, A. Joshi, S. Mankovskii, M. Velez-Rojas, S. Greenspan, and K. Mueller. Collaborative visual analytics using blockchain. In *International Conference on Cooperative Design, Visualization and Engineering*, pp. 209–219. Springer, 2020.

[3] Z. Cutler, K. Gadhave, and A. Lex. Trrack: A library for provenance-tracking in web-based visualizations. In *2020 IEEE Visualization Conference (VIS)*, pp. 116–120. IEEE, 2020.

[4] B. Eidelman. Diff and patch javascript objects. `https://github.com/benjamine/JsonDiffPatch`, 2022. [Online; accessed 23-April-2022].

[5] K. A. Ericsson and A. C. Lehmann. Expert and exceptional performance: Evidence of maximal adaptation to task constraints. *Annual review of psychology*, 47(1):273–305, 1996.

[6] J. Heer. Signals. `https://vega.github.io/vega/docs/signals/`, 2021. [Online; accessed 23-April-2023].

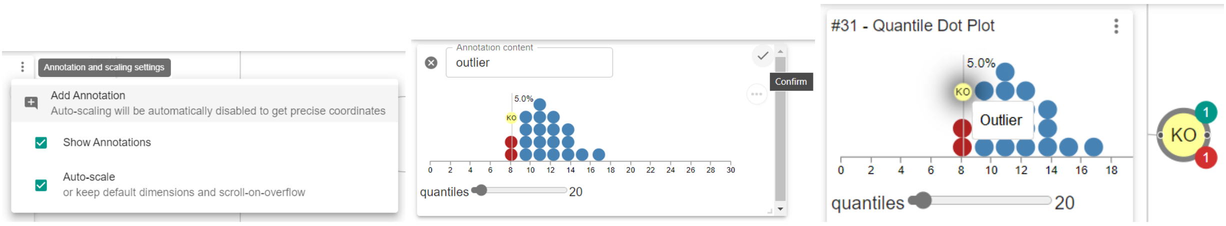
[7] J. Heer, J. Mackinlay, C. Stolte, and M. Agrawala. Graphical histories for visualization: Supporting analysis, communication, and evaluation. *IEEE transactions on visualization and computer graphics*, 14(6):1189–1196, 2008.

[8] M. Herschel, R. Diestelkämper, and H. Ben Lahmar. A survey on provenance: What for? what form? what from? *The VLDB Journal*, 26:881–906, 2017.

[9] J. E. Horvitz. An overview of the art market. *Collectible Investments for the High Net Worth Investor*, pp. 85–117, 2009.

[10] K. Madanagopal, E. D. Ragan, and P. Benjamin. Analytic provenance in practice: The role of provenance in real-world visualization and data analysis environments. *IEEE Computer Graphics and Applications*, 39(6):30–45, 2019.

[11] A. Mathisen, T. Horak, C. N. Klokmose, K. Grønbæk, and N. Elmqvist. Insideinsights: Integrating data-driven reporting in collaborative visual



☑ Show Annotations

☑ Auto-scale
or keep default dimensions and scroll-on-overflow

Annotation content: outlier

Figure 4: Adding Annotations: This figure shows how a user can add an annotation at a specific location on a visualization. The left figure shows the result of clicking on the menu on the visualization, the middle figure shows the user adding an annotation, and the right figure shows the annotation being displayed on the visualization and the corresponding node (labeled KO) with the red circle with a 1 in it.

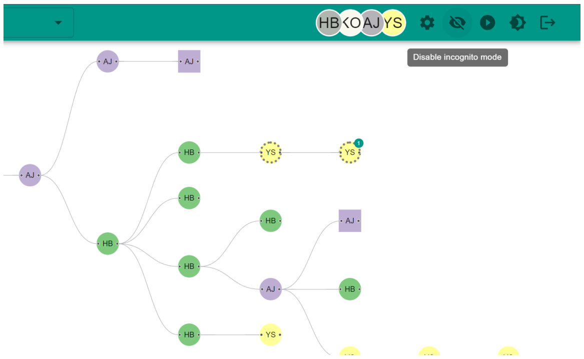


Figure 5: Incognito Mode: This figure shows the appearance of the provenance tree when a user enters incognito mode and interacts with the visualization interface. The nodes connected with dotted edges in the provenance tree are the nodes generated in incognito mode.

analytics. In *Computer Graphics Forum*, vol. 38, pp. 649–661. Wiley Online Library, 2019.

[12] P. Missier, K. Belhajjame, J. Zhao, M. Roos, and C. Goble. Data lineage model for taverna workflows with lightweight annotation requirements. In *Provenance and Annotation of Data and Processes: Second International Provenance and Annotation Workshop, IPAW 2008, Salt Lake City, UT, USA, June 17-18, 2008. Revised Selected Papers 2*, pp. 17–30. Springer, 2008.

[13] R. Neogy. *Synchronized Vega-Lite: designing collaborative visualization*. PhD thesis, Massachusetts Institute of Technology, 2020.

[14] P. H. Nguyen, K. Xu, A. Bardill, B. Salman, K. Herd, and B. W. Wong. Sensemap: Supporting browser-based online sensemaking through analytic provenance. In *2016 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 91–100. IEEE, 2016.

[15] P. H. Nguyen, K. Xu, A. Wheat, B. W. Wong, S. Attfield, and B. Fields. Sensepath: Understanding the sensemaking process through analytic provenance. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):41–50, 2015.

[16] C. North, R. Chang, A. Endert, W. Dou, R. May, B. Pike, and G. Fink. Analytic provenance: process+ interaction+ insight. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems*, pp. 33–36. 2011.

[17] D. Park, M. Suhail, M. Zheng, C. Dunne, E. Ragan, and N. Elmqvist. Storyfacets: A design study on storytelling with visualizations for collaborative data analysis. *Information Visualization*, 21(1):3–16, 2022.

[18] P. Pirolli and S. Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of international conference on intelligence analysis*, vol. 5, pp. 2–4. McLean, VA, USA, 2005.

[19] E. D. Ragan, A. Endert, J. Sanyal, and J. Chen. Characterizing provenance in visualization and data analysis: an organizational framework of provenance types and purposes. *IEEE transactions on visualization and computer graphics*, 22(1):31–40, 2015.

[20] E. D. Ragan and J. R. Goodall. Evaluation methodology for comparing memory and communication of analytic processes in visual analytics. In *Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization*, pp. 27–34, 2014.

[21] E. D. Ragan, J. R. Goodall, and A. Tung. Evaluating how level of detail of visual history affects process memory. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 2711–2720, 2015.

[22] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer. Vega-lite: A grammar of interactive graphics. *IEEE transactions on visualization and computer graphics*, 23(1):341–350, 2016.

[23] J. Zhao, M. Glueck, P. Isenberg, F. Chevalier, and A. Khan. Supporting handoff in asynchronous collaborative sensemaking using knowledge-transfer graphs. *IEEE transactions on visualization and computer graphics*, 24(1):340–350, 2017.