SchemaLine: Timeline Visualization for Sensemaking

Phong H. Nguyen, Kai Xu, Rick Walker, and B. L. William Wong
School of Science & Technology
Middlesex University
London, UK
{p.nguyen, k.xu, r.walker, w.wong}@mdx.ac.uk

***Abstract*—Timeline visualization is an important tool for sensemaking. It allows analysts to examine information in chronological order and to identify temporal patterns and relationships. However, many existing timeline visualization methods are not designed for the dynamic and iterative nature of the sensemaking process and the various analysis activities it involves. In this paper, we introduce a novel timeline visualization, SchemaLine, to address these deficiencies. SchemaLine is designed to group notes into analyst-determined schema, using a layout algorithm to produce compact but aesthetically pleasing timeline visualization, and includes fluid user interactions to support sensemaking activities. It enables interactive temporal schemata construction with seamless integration with visual data exploration and note taking. Our preliminary evaluation results show that the participants found the new method easy to learn and use, and its features effective for the sensemaking activities for which it was designed.**

***Index Terms*—timeline visualization; sensemaking**

I. Introduction

A timeline is a chronicle of events and its visualization is to plot events along the time axis and position them at the time points at which they occur or the ranges over which they last [1]. Sensemaking involves gathering information, representing it in a schema, analyzing that representation, and possibly discovering new knowledge or informing further actions [2]. Pirolli and Card suggest the usefulness of timeline visualizations in the schematization process in their sensemaking model [3]. A timeline helps coordinate events in the dataset chronologically; therefore, it may help reveal temporal relationships and reduce analysts' effort in memorizing them.

Several visual analytics systems integrate timeline visualizations for different purposes. POLESTAR [4] and HARVEST [5] allow users to take notes, define new knowledge, and visualize them in a timeline. Jigsaw [6] provides automatic extraction of entities (people, places, organizations, etc.), and a timeline to organize them. nSpace2 Sandbox [7] allows the creation of multiple bands within the timeline to classify different types of artifacts in the system.

However, the timeline visualizations in these systems suffer from several drawbacks. They either lack an automatic layout [4] or use an overly-simplistic linear layout [7]. As a result, the visualization requires significant effort from users to manually arrange the data items. Schematization [3] is an important process in sensemaking: it involves organizing information into groups or categories, e.g., because it relates to the same person or forms a causal narrative. This can later help analysts form hypotheses about the problem being researched. It is useful to show such information visually on a timeline, so that analysts can study and discover higher-level temporal patterns and relations between groups of events, instead of only those between individual pieces of information.

Sensemaking is a highly iterative process, with each component closely connected to the rest. The Pirolli-Card model [3] depicts it as a hierarchy of sensemaking loops, with the entire process (the top-level loop) divided recursively into smaller sensemaking loops. The Data-Frame model [8] consists of a few interconnected iterative loops, each for a certain type of sensemaking activity. The implication for timeline visualization is that it needs to support the dynamic nature of sensemaking by allowing analysts to interactively create and edit timelines and by providing close integration with other elements of a visual analytics environment, such as visual exploration and argumentation, to support the tight connections between sensemaking tasks. Also, these need to be achieved through intuitive and fluid interaction, so as not to require extra cognitive effort and distract analysts from their current train of thought.

In this paper, we introduce a new timeline visualization, SchemaLine, which is designed to address the aforementioned issues. More specifically, SchemaLine contributes

- a visual design for an interactive timeline that groups notes into schema determined by the analyst,
- an algorithm to automatically generate a compact and aesthetically pleasing visualization of these schema on the timeline, and
- a set of fluid interactions with the timeline to support the sensemaking activities defined in the Data-Frame model.

We conducted a preliminary study to evaluate the effectiveness of SchemaLine in supporting sensemaking. The participants found SchemaLine easy to use and its features effective for the given sensemaking tasks.

II. Related Work

In this section, we consider related work on representing temporal data and producing timelines, before also examining larger visual analytic systems that include timelines in their feature set.

A. *Timeline Visualizations*

A typical example of timeline visualizations is LifeLines [1], a visualization for personal histories, which uses icons to indicate discrete events and thick horizontal lines for continuous ones. When the number of data items is large, they need to be shown collectively rather than individually. The river metaphor [9] is one such method that represents thematic changes in large document collections. Storyline visualizations illustrate the dynamic relationships between entities in a story. The technique was first introduced by Munroe with his hand-drawn visualizations [10]. The visualization summarizes movie plots by depicting each character as a line and each interaction between characters as a converging or diverging bundle of those character lines. Computational layouts have since been introduced to automate the rendering process including work by Tanahashi and Ma [11] and Liu et al. [12].

Visualizing individual events on a timeline is relatively simple; however, showing relationships between events is quite challenging. One approach is to explicitly draw an edge between two related entities as in tmViewer [13]. Edge styles can be used to depict different kinds of relationships; however, drawing a node-link diagram on top of the timeline can cause the visualization to become cluttered even with a small number of events. Another method is to use the concurrent perception ability of humans by using color coding or icons to indicate different groupings. When events are distributed along the timeline, this method introduces a heavy cognitive load for viewers to scan through the entire timespan. Our method uses colored backgrounds and clusters all events belonging to the same group to reduce user effort. Relationships within multiple faceted temporal data are addressed by André et al. in Continuum [14] by using views with different scales and the classic details-on-demand technique to save space. More recently, SemaTime [15] can visualize two different types of relationship: time-dependent (e.g., lives-in) and time-independent (e.g., father-of). SemaTime stacks events vertically and places related events close together. Time-dependent relationships are depicted by using rectangles crossing the relevant common interval of the two events. Time-independent relationships are illustrated by simple arrows.

B. *Timelines within Visual Analytics Systems*

A timeline is commonly integrated into Visual Analytics systems designed for making sense of large and complex datasets including POLESTAR [4], HARVEST [5], Jigsaw [6], [16], and nSpace2 Sandbox [7], [17].

To support sensemaking, timelines are typically used to visualize not raw data, but more meaningful information such as user notes (POLESTAR, HARVEST) or extracted entities (Jigsaw, nSpace2 Sandbox) instead. HARVEST visualizes both raw data and synthesized knowledge in one timeline to allow progressive investigation. However, filtering must be supported to prevent valuable information getting lost among dense data.

Most of the systems use timelines to show notes statically, to present a known story instead of dynamically discovering a hidden story. nSpace2 Sandbox is an exception – it allows users to group related entities into sub-timelines and to alter the entity's date on the timeline if needed. However, one entity cannot be added into multiple timelines, which is necessary when an entity's category is uncertain. Our SchemaLine provides a set of fluid interactions to manipulate notes to build a more semantic schema.

Notes are typically represented using the "sticky-notes" metaphor: a colored rectangle as background with text on top of it. nSpace2 Sandbox provides multiple levels of detail for entities: a short summary, a full article, or even entities of entities. Timelines are commonly visualized as a horizontal axis with notes connecting to the timeline by edges. nSpace2 Sandbox uses a vertical axis timeline as the "diary" metaphor with columns for sub-timelines.

POLESTAR requires manual notes arrangement to fit the display. nSpace2 Sandbox uses a simple linear layout to organize entities, thus entities with nearby dates will overlap on the timeline. Our layout algorithm produces an aesthetically pleasing visualization that avoids this issue while still providing easy note manipulation.

Timelines are often used as an extra view, coordinated with the whole system. Jigsaw provides a reasoning space called Tablet, where a timeline can be added. nSpace2 Sandbox also introduces a separate component called Timeline view. Even though entities from data space can be dropped into timeline space, it may introduce a heavy cognitive load for users to switch between two working spaces. In the evaluation of this paper, we integrate SchemaLine into an existing system seamlessly to provide concurrent exploration and sensemaking with data.

III. Sensemaking with Timeline Visualizations

SchemaLine is intended to support tasks in sensemaking of temporal data and is influenced by the well-established sensemaking model proposed by Pirolli and Card [3]. This model organizes the sensemaking process into two loops: the foraging loop, which involves searching, extracting and organizing information; and the sensemaking loop, which involves building schema, creating and testing hypotheses, and presentation. In this model, *schematization* serves as a bridge connecting the foraging loop and the sensemaking loop. It is a crucial step in converting raw evidence to rational explanations. Pirolli and Card suggest that the schematization process should be supported by a computer-based tool that coordinates events in the dataset to reveal relationships between them and to leverage analysts' effort in memorizing them [3]. As a result, we decided to investigate timeline visualization support for sensemaking. A timeline can not only reveal the temporal relationships among the findings, but also have a considerable impact on how easily they can be understood: when Pennington and Hastie [18] studied the impact of evidence presentation order on juror decision making, they found that information was easier to understand when presented in chronological order and thus had a significant impact on jurors' decisions.

We find that the cognitive processes in the schematization process are well elaborated through different sensemaking activities in the Data-Frame model proposed by Klein et al. [8]. These sensemaking activities are *Connect data to a frame*, *Elaborate a frame*, *Question a frame*, *Preserve a frame*, and *Reframe*. Sensemaking activities begin when a surprise, unexpected event with respect to our prior knowledge appears. The analyst forms an initial account for the unexpected event by connecting some evidence. In the Data-Frame model's terminology, the analyst tries to match some data to create an initial frame. When encountering new data, the analyst can either add it to the frame to elaborate the frame (if it fits to the frame) or remove existing data (if it cannot fit the frame anymore). The analyst starts questioning the frame when they detect inconsistencies between data, or poor quality data in the frame. Then, they need to decide between preserving the frame by looking for more data, or reframing it by comparing it with other frames, or seeking a completely new frame. Because of the various and detailed sensemaking activities surrounding the *frame* in the Data-Frame model, we decide to support all these five activities in our the timeline visualization through fluid user interactions. The terms 'schema' and 'frame' are used to refer to the same concept throughout this paper.

IV. Visual Encoding

A. Event Representation

An event is represented by a rounded rectangle with its left side aligned with the event's time on the timeline. To reduce cluttering, events are not constantly connected by lines to their corresponding points on the timeline. Instead, when the mouse is over an event, its time point on the timeline is highlighted. A short textual summary is rendered inside the rectangle to summarize the event. To address the scalability of long summaries, we assign a maximum width to event rectangles and trim excess text. The full content will only be displayed when the note is hovered over. All events have a uniform height to give a consistent overall appearance, especially when they are connected to form a schema (Section V-B). Quite often, events are categorical data. For example, in news reports, an article can be classified into sport, fashion or both. SchemaLine adds a small rectangle in each event to color-code its categorization. Eight different colors are supported, which are chosen from qualitative colors – Set 1 of ColorBrewer [19]. All other categories besides eight of the most popular ones will share the same color to address the limitation of the small number of distinguishable colors. We plan to combine colors with other indicators such as texture to increase the number of differentiated keywords in the future work. The number of maximum categories that an event can belong to is configurable to adapt the dataset characteristics. As in Fig. 1, maximum three themes of an event can be displayed.

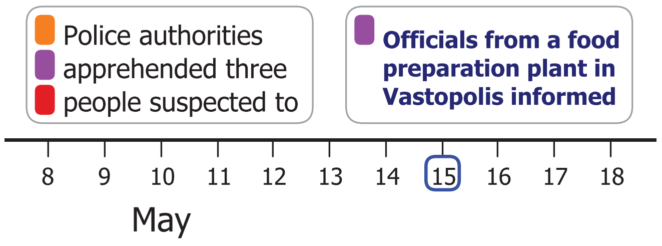


Fig. 1. Events are represented as rounded rectangles with a uniform height and limited width aligned at their corresponding time points. The 15th-May event is highlighted. On the left side of the event rectangle, small color-coded rectangles indicate the event's groupings.

In SchemaLine, the timeline is shown as a horizontal axis at the bottom of the display. Its starting and ending points change dynamically to cover the time span of all events. The timeline consists of two temporal scales. These two scales can also be changed dynamically according to the displayed events. For example, they change from "month/day" to "year/month" to accommodate large interval increases.

B. Schema Representation

After discovering a number of relevant events or pieces of evidence, the analyst starts combining them to form a *schema*. A schema is a set of related events that are connected to each other in a certain way. For example, a schema might contain all events about a particular person. Multiple schemata can be composed in SchemaLine as shown in Fig. 2.

We consider several design options to connect events within a schema such as using colored/shaped icons or node-link diagrams. However, they all have some drawbacks as discussed in the Related work (Section II). Computational methods that allow visualize a large number of events with different themes such as ThemeRiver [9] do not work either because individual events and interactions are more essential in SchemaLine. Also, it should be easy to follow events within a schema in temporal order. We decided to visualize each schema as a colored stripe, which is inspired by Munroe's hand-drawn visualization [10]. A character line in Munroe's work connects all events happened to that character. Similarly, our schema is a color stripe connecting all events belonging to it. Instead of using a thin line, we use a path with unique width (an event's height) to make enough space to display the event's summary text and allow interaction with individual notes. A rectilinear path is employed to provide a nice visualization rather than direct connection between events.

V. Algorithm

The process of generating schemata has two main steps. First, the layout of the schemata is generated and then its outline is computed based on the layout information.

A. SchemaLine Layout

The algorithm that produces the layout of schemata and events in SchemaLine aims to produce a compact and aesthetically pleasing visualization that also meets the following criteria:

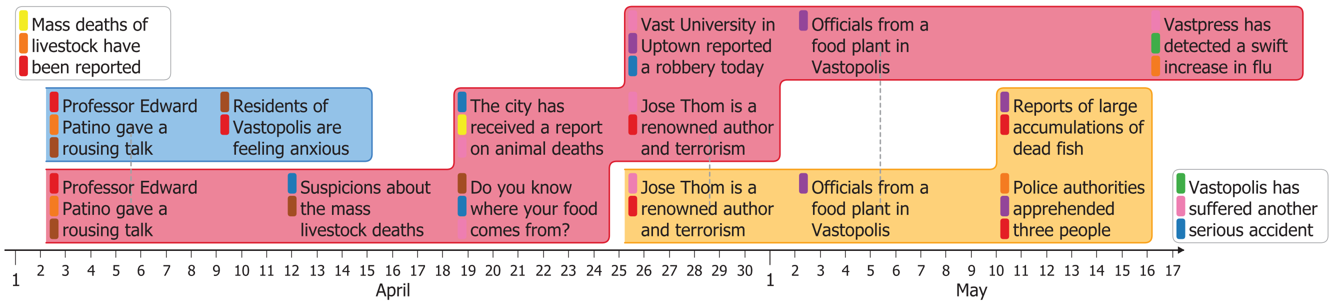

Mass deaths of livestock have been reported
Professor Edward Patino gave a rousing talk
Residents of Vastopolis are feeling anxious
Professor Edward Patino gave a rousing talk
Suspicions about the mass livestock deaths
The city has received a report on animal deaths
Do you know where your food comes from?
Vast University in Uptown reported a robbery today
Jose Thom is a renowned author and terrorism
Officials from a food plant in Vastopolis
Jose Thom is a renowned author and terrorism
Officials from a food plant in Vastopolis
Vastpress has detected a swift increase in flu
Reports of large accumulations of dead fish
Police authorities apprehended three people
Vastopolis has suffered another serious accident
April
May

Fig. 2. SchemaLine: each piece of text is an analyst note, positioned along the time axis at when the event happened. Related notes are linked together to form a "schema" or "frame". There are three frames in this example represented as colored rectilinear paths. Small color-coded rectangles on the left side of notes are "categories".

$C1$ The preferred horizontal position of an event is its corresponding time on the timeline.

$C2$ An event can be shifted horizontally by a limited amount to improve the layout; however, the relative order between events must be maintained.

$C3$ There is no event/event, event/schema, or schema/schema overlap.

In summary, the layout algorithm consists of the following four steps (Fig. 3):

1) Order the schemata such that those that share events are next to each other as much as possible;
2) Generate the relative position of events within a schema;
3) Place schemata bottom up following the order computed in the first step as compactly as possible;
4) Add the remaining events that do not belong to any schema.

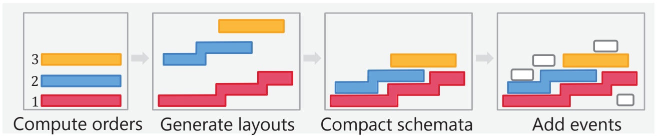


Fig. 3. The SchemaLine algorithm: First, the order of schemata is computed. Second, the layout of each schema is generated independently. Third, schemata are stacked together to save display space without changing order. Finally, events that do not belong to any schema are added.

1) Schema Orders: It is not always possible to have schemata that share events placed next to each other. For example, if three schemata all share events with each other, then in any order two of them will always be separated by the third schema. Our algorithm uses a strategy that prioritizes pairing of schemata according to how many events they share. To do this, we map the problem to graph path finding as below. Given a set of schemata S, we create an undirected graph $G = (V, E)$, where each vertex v_i represents a schema $s_i \in S$. The weight of an edge e_{ij} is the number of events shared by schemata s_i and s_j. Finding a schema order with the maximum number of shared nodes placed next to each other becomes finding a path with maximum weight connecting all vertices in G. This classic longest path finding problem is NP-hard. The number of schemata we plan to support is at most eight due to the limitation of the small number of colors that human can distinguish. Therefore, we simply use brute-forte algorithm to find the path.

2) Individual Schema Layout: The second step of the algorithm produces the layout of each schema. Events shared by multiple schemata are replicated for each of them; therefore, the layout of each schema can be generated independently. Events within a schema are sorted chronologically so that they can be added from left to right. The algorithm works by adding one event at a time: the new event will stay at the same horizontal level as the previous one if it can, otherwise it will move up one level. When a event n_i has the same time as the previous n_{i-1}, it needs to be moved up one level because they must have the same x-coordinate. Otherwise, if n_i intersects with n_{i-1}, an attempt is made to shift n_{i-1} to the left to make space for n_i as discussed below. If the shifting is successful, the event stays in that level; otherwise, the event needs to move up one level.

Shifting Events: To address the issues of scalability and efficient use of space, accuracy of the event position can be sacrificed. For each event, its x-coordinate is initially set at event time ($C1$). Then, events can be shifted horizontally to the left by a limited amount, to make room for events that are after it temporally. An event can be rendered at the non-accurate position scaled with its time. However, to keep the event close to the accurate position, we set the maximum distance that a event can shift to its width. As a result, the event still overlaps with its time point on the timeline and provides reasonable indication to viewers of its true position. When shifting, it is crucial to maintain the relative order between the shifting event and other events ($C2$). For example, if event n_i was to the left of n before the shifting, it should remain on the left afterwards. Another important condition is that there is no intersection with any other event after the shifting ($C3$). An illustration of this algorithm is shown in Fig. 4.

3) Schemata Compact: In the third step, the algorithm stacks schemata in the order computed in the first step to produce a compact visualization. For example, if two schemata cover non-overlapping time ranges, they can be placed in the same level to save display space.

The group of schemata that share events is added to the

SchemaLine first. Their relative ordering top to bottom is fixed and each schema is pushed towards the bottom as much as possible to save display space. After this, schemata without any shared event are added, again from bottom up to find the lowest level possible.

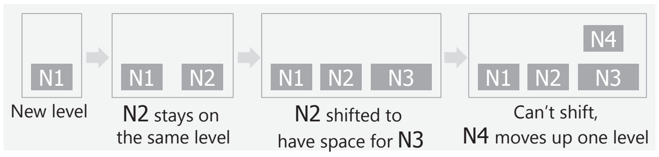

N1
New level
N1 N2
N2 stays on the same level
N1 N2 N3
N2 shifted to have space for N3
N4
N1 N2 N3
Can't shift, N4 moves up one level

Fig. 4. Schema layout algorithm. Four events $N1$, $N2$, $N3$, $N4$ will be added to the schema in chronological order. $N1$ is positioned at its accurate event time. $N2$ can stay in the same level as $N1$ because it does not intersect with $N1$. $N3$ intersects with $N2$ but the intersection width is small enough so that $N2$ can be shifted to the left to let $N3$ stay in the same level as well. However, $N4$ needs to move up one level because the width of its intersection with $N3$ is longer than that of $N1$ and $N2$, i.e., they cannot be shifted.

4) Non-schema Events: This last step allocates events that do not belong to any schemata. Events are sorted chronologically so that they are added to the SchemaLine from left to right. The ideal x-position is the event time, but an event can be shifted as described in Section V-A2. An event always begins at the lowest level and moves upward until there is enough space for it: that is, until it does not intersect with any other schema or events after possible horizontal shifting.

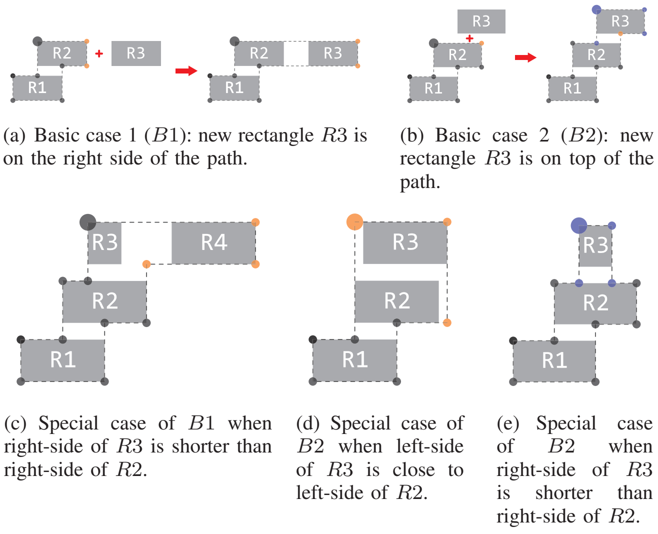
B. SchemaLine Outline

In this section, we describe the algorithm to produce a polygonal outline covering all the event rectangles within a schema. We decided to use only horizontal or vertical line segments to keep the outline simple. The polygonal path P_n of a schema that contains n event rectangles $R_1, R_2, ..., R_n$, ordered from left to right, is determined as follows:

$$P_n = \begin{cases} R_1, & n = 1 \\ P_{n-1} \oplus R_n, & n > 1 \end{cases},$$

where $\oplus$ is the *merge operator* that merges a polygonal path and a rectangle into a new polygonal path.

A polygonal path is simply represented as an array of vertex coordinates. As the above formula demonstrates, this array will be incrementally extended by adding each rectangle individually. As described in the schema layout algorithm (Section V-A2), when adding a new event into an existing schema, the event either has the same level as the previous event of the schema or moves up one level. Polygonal path extension is simple; however, it needs to extensively cover all possible cases to produce a nice path. Fig. 5 shows five different cases that need to be considered when merging a polygonal path and a rectangle.



(a) Basic case 1 ($B1$): new rectangle $R3$ is on the right side of the path.

(b) Basic case 2 ($B2$): new rectangle $R3$ is on top of the path.

(c) Special case of $B1$ when right-side of $R3$ is shorter than right-side of $R2$.

(d) Special case of $B2$ when left-side of $R3$ is close to left-side of $R2$.

(e) Special case of $B2$ when right-side of $R3$ is shorter than right-side of $R2$.

Fig. 5. Five possible cases when a new rectangle into the polygonal path. Big circle indicates the pivot vertex of the path (top-left corner of the last rectangle). Orange circles indicate updated vertices, and blue circles indicate newly added vertices of the polygonal path.

After producing a rectilinear path, the bends are made rounded to create a pleasing visualization (Fig. 2). The path is filled with the same stroke color but less transparency to make the border pop-out with a darker hue. The beginning of the path does not have the border to indicate that the path is open on that side and the reader should follow this direction.

VI. Sensemaking with SchemaLine

SchemaLine is designed to support all five sensemaking activities in Data-Frame model through fluid user interactions. Following the design guidelines for fluidity proposed by Elmqvist et al. [20], SchemaLine's interactions

- use smooth animated transitions between states,
- provide immediate visual feedback on interaction, and
- use direct manipulation of visual representations.

Sensemaking activities in Data-Frame model involve two different types of entities: *data* and *frame*. We allow direct manipulation of visual representations of data and frame, instead of invoking menus and buttons to perform actions. The first sensemaking activity in the Data-Frame model is to **construct a new frame** by connecting relevant data. It can be performed in SchemaLine by *dragging one event and dropping it onto another event*. A *plus* icon and a *dashed rectangle* surrounding the two events are displayed to indicate that a new frame will be created. When dropping the event, a color stripe representing a frame will be formed by connecting these two events, and a smooth animated transition is used to improve user perception.

Besides dropping an event on top of another event, the user can drop it onto the color stripe to add that event to an existing frame (**elaborate a frame**). Conversely, the user can drag an event belonging to a frame and drop it onto the void space to remove it from the frame (**preserving a frame**). Appropriate informative feedback is displayed, *plus* icon for addition and *minus* icon for subtraction, and a smooth animated transition is used to improve user perception. Fig. 6 shows an example of adding an event into a frame.

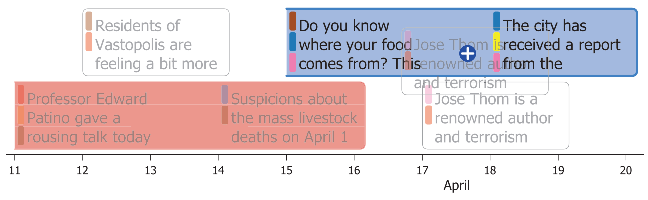

Residents of Vastopolis are feeling a bit more
Do you know where your food comes from? This
The city has received a report from the
Jose Thom is a renowned author and terrorism
Professor Edward Patino gave a rousing talk today
Suspicions about the mass livestock deaths on April 1
Jose Thom is a renowned author and terrorism
11
12
13
14
15
16
17
18
19
20
April

Fig. 6. Each frame is represented as a colored stripe. Dropping an event onto the blue stripe means adding that event into the blue frame to elaborate it.

Questioning a frame occurs when the user encounters inconsistencies in data within a frame. The temporal distribution of events in the frame may suggest some concerns about the validity or completeness of the frame. For example, if a frame about one person contains many events in January and March, but no events are found in February, then it may be inferred that there could be some data missing. The analyst can mark a suspected event by right-mouse double-clicking on it. Red color text is used to indicate that the event needs more investigation.

Dragging an event from one frame to another frame will remove it from the old frame and add it to the new frame. However, holding *Control* key when dropping will instead copy the event to the new frame. This interaction allows the analyst to duplicate events to create several similar frames and compare them (**comparing frames**). When two frames are selected, they will be moved closer together to allow easy comparison, irrespective of the frames ordering generated by the layout algorithm. The user can drag an entire frame and drop it onto another frame to merge all events together. The user can also drop the frame onto the void space to take apart the frame and release its events. This interaction is useful when the user thinks that the frame is completely wrong and wants to construct a new frame (**reframing**).

Other interactions with events are also designed to be intuitive. Left-mouse double-clicking on an event opens its full content. Dragging an event with the right mouse button can change the event's date. This feature is useful because the report date is not always the date when the event actually occurred; for example, "yesterday there was a bomb attack in ABC". Dragging an event outside the boundary of the timeline will remove it from the system (with *remove* icon as informative feedback).

Once any change is made on SchemaLine, such as moving an event from one frame to another, an animation is shown of smooth transition between the changes to help analyst update their "mental map". To achieve this, the layout algorithm (Section V-A) computes the new event rectangle locations. Then, the outline algorithm (Section V-B) runs at every step of the interpolation between the old and the new locations to produce intermediate polygon paths based on the updated event locations.

VII. Evaluation

We first discuss the integration of SchemaLine into an existing visual analytics system, and then conduct an evaluation of SchemaLine's usefulness in supporting sensemaking of temporal data.

A. An Application of SchemaLine

To evaluate the usefulness of SchemaLine, we integrated it into an existing sensemaking system that also follows the Pirolli-Card model. We choose our own research framework, INVISQUE [21], an INteractive VIsual Search and QUery Environment, so that the integration can be done at the code level. Following the Pirolli-Card model, SchemaLine uses the output of the Read & Extract process, evidence files, as the input.

INVISQUE provides keyword search capability to address the Search & Filter process. The search results are shown as a cluster of *index-cards*, each representing a document with selected information. For example, an index-card might have title, publishing date, and the first lines of the full article for a news report. The analyst can take notes while reading it. Notes will be saved and rendered at the bottom of the index-card. Notes are considered as *evidence files* in the Pirolli-Card model and used as the input of SchemaLine. To minimize the analyst's effort, when they finishes entering a note, both the note and its associated document are passed to SchemaLine and automatically added to or updated on the timeline. Left-mouse double-clicking on the note will open the original document in index-card metaphor. After examining search results, the analyst can minimize the cluster and leave only the search term visible, to make more screen space available for new searches but still be able to recall them back later. We color-code the minimized clusters with the aforementioned small rectangles inside the note rectangle. This helps indicate the provenance of each note and may support sensemaking activities at a later stage.

B. Case Studies

Evaluating the usefulness of SchemaLine in supporting sensemaking is challenging. It is categorized as *evaluating visual data analysis and reasoning* – one of seven scenarios in the information visualization empirical studies by Lam et al. [22]. Because of the difficulties of this evaluation type, such as the fluidity and various approaches used by analysts and the quantification of the analysis results, evaluations are typically case studies with realistic datasets and domain experts as participants.

We used the task from Mini Challenge 3 of the VAST Challenge 2011, which requires the participants to identify any potential criminal activities from the given dataset. INVISQUE with SchemaLine integrated was used in the study. This dataset was chosen because the solution was provided and well-tested by the community. The original dataset contains four thousand news reports, many of which are over 500 words long. A pilot study showed that it was difficult for the participant to find any answers even after trying for a long time. The reason could be INVISQUE does not support text-mining features such as entity extraction, which is crucial in analyzing a large document collection. The goal of the evaluation is to assess how SchemaLine can support INVISQUE in sensemaking, not to assess INVISQUE itself so, in the actual study, we reduced the dataset to contain only 36 documents that were manually added into the dataset as part of the ground truth so that participants could complete the task with reasonable time and effort but without affecting the goal of the evaluation. Five criminal activities are embedded into those documents including food poisoning (13 documents), hacking (3), dirty bomb (6), arms trafficking (4), and money laundering (3) together with 7 isolated cases, which are not considered as correct solutions.

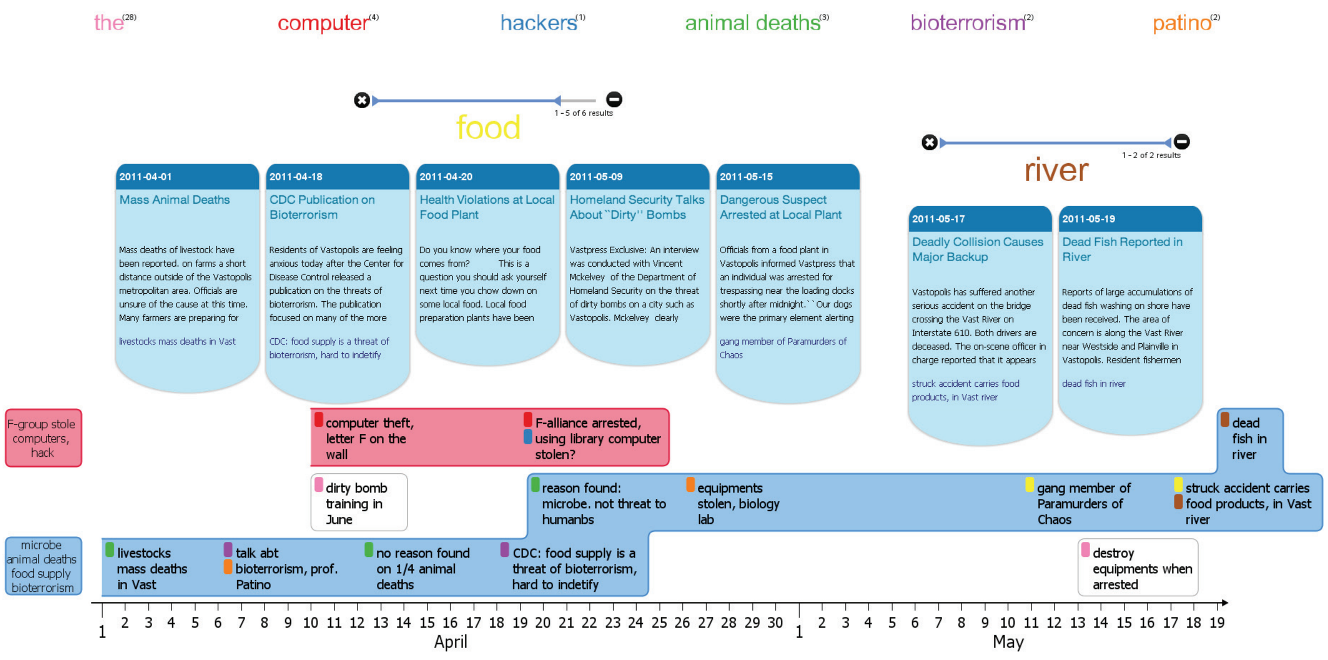
SchemaLine is designed for general-purpose usage. We tried to recruit participants with varying backgrounds, and conducted three case studies with a graduate student in visual analytics (surrogate for visualization expert, **P1**), a graduate student in law (surrogate for domain expert, **P2**), and a graduate student in networking (neutral background participant **P3**). A 10-minute-training session was given before an one-hour main task to help participants become familiar with the basic functions of INVISQUE and the set of interactions that SchemaLine offers to group relevant notes. After finishing their analyses, participants reported the criminal activities they had discovered, together with the supporting evidence. The main methods of these case studies were observations and interviews, which focused on how SchemaLine facilitates the participant in finding the answers in the context of the five sensemaking activities in the Data-Frame model it is designed to support.

1) Case Study 1 – Visual Analytics Graduate Student: Participant **P1** began searching for "bomb", read, took notes and continued searching for a more detailed keyword "dirty bomb". He used drag-and-drop interaction to link his three notes of "dirty bomb" documents together (*construct a new frame*). Then, he searched for "Network of Dread", which was mentioned in one document as the author of the dirty bomb attack. He took notes on the new returned document and dropped it onto the color stripe of dirty bomb (*elaborate a frame*). While investigating, he encountered an article about a man carrying a frozen turkey having wires coming out it, which was suspected as a bomb. At first, he dropped the "turkey bomb" note onto the "dirty bomb" stripe. Then, he wondered "Is it a real bomb?". After thinking for a while, he removed it out of the stripe (*preserve a frame*), which was a correct decision when checking against the solution.

2) Case Study 2 – Law Graduate Student: Participant **P2** took an overview step before searching. He quickly looked at all 36 document titles to have a glimpse of the dataset as well as to detect potential search keywords. He began searching "animal deaths", read, took notes and grouped them together (*construct a new frame*). He was happy with the evidence he found for that crime and switched to read another interesting article "Library Computer Left" he came across. From that, he searched for several related terms such as "computer" and "hackers". He figured out that a group called "F-alliance" stole computers from the library and attempted to hack a bank. He dropped the "computer stolen" note on top of the "bank hacking" note to form a new explanation for the case (*construct a new frame*). He found another article related to hacking but he said "I won't drop it to this group because it's just an announcement from the government about potential threats" (*preserve a frame*). During further investigation, he created another group of notes related to "bioterrorism" and "Prof. Patino". Then, when figuring out that the reason of the mass deaths is a spore-forming microbe, which is also mentioned in Prof. Patino's talk, he dragged the new group and dropped it onto the "animal deaths" group to combine all notes together because they are related (*merge frames*). Observing the event orders in the new group on the timeline, he said "The equipment of Patino was stolen after the animal deaths report, so they couldn't be used in that case. This is the group of potential threat in using bioterrorism" (*elaborate a frame*). Fig. 7 shows the computer screen of **P2** when he reported his findings.

3) Case Study 3 – Computer Network Graduate Student: Participant **P3** searched for a few keywords related to criminal activities before investigating the search results such as "bomb", "terrorism", "money" and "hack". In a similar fashion to other participants he took notes when reading, and grouped related notes together by dragging one note and dropping it on top of another note (*construct a new frame*). After finding a crime about money laundering, he read articles from "terrorism" search results. Then, he followed the article content to search for relevant information such as "Paramurderers of Chaos" – a terrorist group. During further investigation, similar to **P2**, he also combined two groups of notes – "Paramurderers of Chaos" and "food supply", together when discovering evidence linking the two groups (*merge frames*). When representing his findings, he shared that SchemaLine prompted him to look for missing information. "I noticed the gap between these two events [pointing to the timeline]; then I knew I probably missed something there" (*question a frame*).

4) Participant Results and Discussion:: **P1** found the "dirty bomb" attack with 4/6 pieces of evidence. **P2** found the "hacking" case with 2/3 pieces of evidence, and the "food poisoning" case with 9/13 pieces of evidence. **P3** also found the "food poisoning" case with 6/13 pieces of evidence, and a perfect 3/3 pieces of evidence in the "money laundering" case. **P1** took many notes in documents related to the "food poisoning" case; however he could not link them together because "I'm not familiar with bio-attack so I couldn't think of it as a threat". All participants extensively used SchemaLine to group related notes with different types of relationship: a group of "bioterrorism" articles, a group of more specific criminal activity like "dirty bomb" or "money laundering", a group of people like "Paramurderers of Chaos", and a group of a specific person like "Prof. Patino". Another benefit of using SchemaLine is that all participants are very confident when presenting their analyses. **P3** even opened the original


the(28)
computer(4)
hackers(1)
animal deaths(3)
bioterrorism(2)
patino(2)
food
1 - 5 of 6 results
river
1 - 2 of 2 results
F-group stole computers, hack
computer theft, letter F on the wall
F-alliance arrested, using library computer stolen?
dirty bomb training in June
reason found: microbe. not threat to humanbs
equipments stolen, biology lab
gang member of Paramurders of Chaos
struck accident carries food products, in Vast river
dead fish in river
microbe animal deaths food supply bioterrorism
livestocks mass deaths in Vast
talk abt bioterrorism, prof. Patino
no reason found on 1/4 animal deaths
CDC: food supply is a threat of bioterrorism, hard to indetify
destroy equipments when arrested
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
April
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
May

2011-04-01
Mass Animal Deaths
Mass deaths of livestock have been reported. on farms a short distance outside of the Vastopolis metropolitan area. Officials are unsure of the cause at this time. Many farmers are preparing for
livestocks mass deaths in Vast

2011-04-18
CDC Publication on Bioterrorism
Residents of Vastopolis are feeling anxious today after the Center for Disease Control released a publication on the threats of bioterrorism. The publication focused on many of the more
CDC: food supply is a threat of bioterrorism, hard to indetify

2011-04-20
Health Violations at Local Food Plant
Do you know where your food comes from? This is a question you should ask yourself next time you chow down on some local food. Local food preparation plants have been

2011-05-09
Homeland Security Talks About ``Dirty'' Bombs
Vastpress Exclusive: An interview was conducted with Vincent Mckelvey of the Department of Homeland Security on the threat of dirty bombs on a city such as Vastopolis. Mckelvey clearly

2011-05-15
Dangerous Suspect Arrested at Local Plant
Officials from a food plant in Vastopolis informed Vastpress that an individual was arrested for trespassing near the loading docks shortly after midnight.``Our dogs were the primary element alerting
gang member of Paramurders of Chaos

2011-05-17
Deadly Collision Causes Major Backup
Vastopolis has suffered another serious accident on the bridge crossing the Vast River on Interstate 610. Both drivers are deceased. The on-scene officer in charge reported that it appears
struck accident carries food products, in Vast river

2011-05-19
Dead Fish Reported in River
Reports of large accumulations of dead fish washing on shore have been received. The area of concern is along the Vast River near Westside and Plainville in Vastopolis. Resident fishermen
dead fish in river

Fig. 7. A reproduced image from the video record of participant **P2** when he reported his findings. Top: a trail of his keyword searches, collapsed after being read. Middle: search results in index-card metaphor. Bottom: two schemata containing notes as supporting evidence of criminal activities he found.

document (double-clicking on the note) several times to highlight the relevant text when reporting to the observer to show his strong evidence. **P1** had a scenario about *airport attack* with only two pieces of evidence. He said that he was not sure about that and he did not think it was correct. Indeed, it was not mentioned in the solution. It proved that the participants justified evidence a lot to provide a reliable answer when using SchemaLine.

In summary, all participants liked the feature of automatically adding notes to the timeline so that they did not need to remember the notes. **P1** thought that he would have problem if the system did not support that: "I can remember what happened but it was difficult to remember when it happened". They found that it was helpful to build a scenario with SchemaLine because the information is organized chronologically, which is the order when events actually happened. **P2** shared that he read the news about the robbery at Vastopolis university and the Prof. Patino's talk about bioterrorism. He did not have any insight at that time. However, when looking at his two notes on the timeline, he realized that the order of the two events was the other way around. Then, he thought that Prof. Patino's description of extremely expensive equipment in his lab could be the rationale of the robbery. All participants commented that the interactions to edit frames are very intuitive. **P1** even said "I think I don't even need training and still can figure out how it works". **P3** liked the transition effect when adding or removing notes because "it helped me to understand what is going on". These case studies showed the usefulness of SchemaLine in supporting sensemaking as well as the intuitiveness and effectiveness of SchemaLine's interactions in performing those sensemaking activities.

VIII. Conclusions and Future Work

In this paper, we introduced a new timeline visualization, SchemaLine, which is designed to support sensemaking. More specifically, it facilitates the schematization process in the Pirolli-Card model and targets all sensemaking activities in the Data-Frame model. The SchemaLine layout algorithm produces simple, compact, but aesthetically pleasing timeline visualizations. It replaces menu and buttons with fluid user interactions to perform all necessary tasks, and can be integrated within larger visual analytic systems. Our preliminary evaluation suggests that the design of SchemaLine is supportive of sensemaking tasks. It was clearly a helpful aid to users in analysis of the scenario, as evidenced by their usage patterns and feedback.

As future work, a more formal evaluation would be beneficial – perhaps even following integration of SchemaLine into a number of different systems, to allow the specific effect to be separated from the rest of the system. In terms of design of the SchemaLine itself, there are a number of improvements that could be added. Shared events between frames could be better visualized (at present, the event is simply duplicated). There are also obvious issues with scalability: while the timeline will scale comparatively well with number of events, it will scale badly with number of frames, since the set of effective qualitative colors is quite small. Other cues such as texture or line style may help with this problem, but to discover this will require further experimentation.

References

[1] C. Plaisant, B. Milash, A. Rose, S. Widoff, and B. Shneiderman, "LifeLines: visualizing personal histories," in *ACM Conference on Human Factors in Computing Systems*, Apr. 1996, pp. 221–227.

[2] S. K. Card, J. D. Mackinlay, and B. Shneiderman, *Readings in information visualization: using vision to think*. Morgan Kaufmann Publishers Inc., Jan. 1999.

[3] P. Pirolli and S. Card, "The Sensemaking Process and Leverage Points for Analyst Technology as Identified Through Cognitive Task Analysis," in *Conference on Intelligence Analysis*, 2005.

[4] N. J. Pioch and J. O. Everett, "POLESTAR - Collaborative Knowledge Management and Sensemaking Tools for Intelligence Analysts," in *ACM International Conference on Information and Knowledge Management*, Nov. 2006, pp. 513–521.

[5] D. Gotz, M. Zhou, and V. Aggarwal, "Interactive Visual Synthesis of Analytic Knowledge," in *IEEE Symposium on Visual Analytics Science And Technology*. IEEE, Oct. 2006, pp. 51–58.

[6] C. Gorg, Z. Liu, and J. Stasko, "Reflections on the evolution of the Jigsaw visual analytics system," *Information Visualization*, Jul. 2013.

[7] "Timelines in the nSpace2 Sandbox," http://www.oculusinfo.com/assets/nspace2videos/TimelinesInTheSandbox.mp4, 2012.

[8] G. Klein, J. K. Phillips, E. L. Rall, and D. A. Peluso, "A Data-Frame Theory of Sensemaking," in *Expertise out of context: Proceedings of the sixth international conference on naturalistic decision making*, R. R. Hoffman, Ed. Mahwah, NJ: Lawrence Erlbaum Associates, 2003, pp. 113–155.

[9] S. Havre, E. Hetzler, P. Whitney, and L. Nowell, "ThemeRiver: visualizing thematic changes in large document collections," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 1, pp. 9–20, 2002.

[10] R. Munroe, "Xkcd #657: Movie narrative charts," http://xkcd.com/657, 2009.

[11] Y. Tanahashi and K.-L. Ma, "Design Considerations for Optimizing Storyline Visualizations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2679–2688, Dec. 2012.

[12] S. Liu, Y. Wu, E. Wei, M. Liu, and Y. Liu, "StoryFlow: tracking the evolution of stories," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2436–2445, 2013.

[13] V. Kumar, R. Fur, and R. B. Allen, "Metadata Visualization for Digital Libraries : Interactive Timeline . Editing and Review," in *ACM conference on Digital libraries*, 1998, pp. 126–133.

[14] P. André, M. L. Wilson, A. Russell, D. A. Smith, A. Owens, and M. Schraefel, "Continuum: designing timelines for hierarchies, relationships and scale," in *ACM Symposium on User Interface Software and Technology*, Oct. 2007, pp. 101–110.

[15] C. Stab, K. Nazemi, and D. W. Fellner, "SemaTime - Timeline Visualization of Time-Dependent Relations and Semantics," *Advances in Visual Computing*, pp. 514–523, 2010.

[16] J. Stasko, C. Gorg, Z. Liu, and K. Singhal, "Jigsaw: Supporting Investigative Analysis through Interactive Visualization," in *IEEE Symposium on Visual Analytics Science and Technology*, 2007, pp. 131–138.

[17] W. Wright, D. Schroh, P. Proulx, A. Skaburskis, and B. Cort, "The sandbox for analysis: concepts and methods," in *ACM Conference on Human Factors in Computing Systems*, 2006, pp. 801–810.

[18] N. Pennington and R. Hastie, "Cognitive theory of juror decision making: The story model," *Cardozo L. Rev.*, vol. 13, p. 519, 1991.

[19] M. Harrower and C. a. Brewer, "ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps," *The Cartographic Journal*, vol. 40, no. 1, pp. 27–37, Jun. 2003.

[20] N. Elmqvist, A. V. Moere, H.-C. Jetter, D. Cernea, H. Reiterer, and T. Jankun-Kelly, "Fluid interaction for information visualization," *Information Visualization*, vol. 10, no. 4, pp. 327–340, Aug. 2011.

[21] W. Wong, R. Chen, N. Kodagoda, C. Rooney, and K. Xu, "INVISQUE: intuitive information exploration through interactive visualization," in *Extended Abstracts on Human factors in computing systems*, May 2011, pp. 311–316.

[22] H. Lam, E. Bertini, P. Isenberg, C. Plaisant, and S. Carpendale, "Empirical Studies in Information Visualization: Seven Scenarios," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 9, pp. 1520–1536, Nov. 2012.