Eurographics Conference on Visualization (EuroVis) 2014
H. Carr, P. Rheingans, and H. Schumann
(Guest Editors)

Volume 33 (2014), Number 3

ActivityVis: Abstraction and Visualization of User Email Activities

Submission number: 238

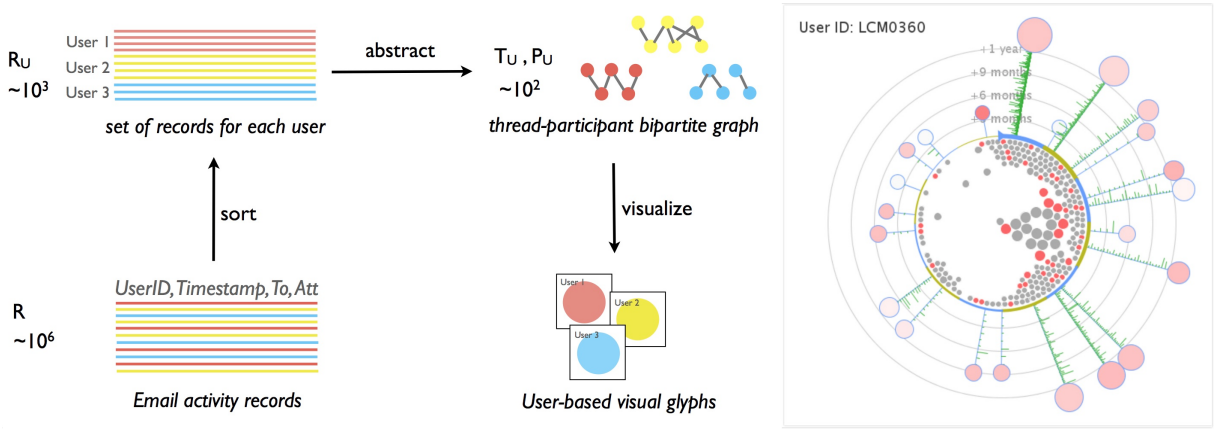

Figure 1: *The flow chart on the left illustrates our data transformations. It summarizes a collection of email records which are eventually abstracted into thread-participant bipartite graphs for visualization. The right figure shows the overall visualization of a user's email activity over a year: Sizes of attachments in all email records are plotted as tangential (green) bars on radial line segments representing threads, which are themselves laid out on a circular timeline. Inside it is the layout of a node group representing the set of people contacted by the user.*

Abstract

Organizational email services help keep an account of the communication activities in workplace, thus offering tremendous opportunity for understanding activity patterns. In this paper, we address the data abstraction and visualization of the outbound email history of individual users. We demonstrate it using a synthetic organizational network which contains millions of email records simulated according to read-world email activity statistics. We identify four research challenges and provide the rationale behind our choices of abstraction and visualization. We describe a self-contained software prototype, ActivityVis. It supports user-based visualizations of outbound email activities, and tabular interface for filtering, searching, comparing and make associations among a set of users. To improve clarity, we use two techniques: (1) force-directed node packing for visual clutter reduction, and (2) angular fisheye distortion for zooming into details on a circular timeline. Additionally in this paper, we report case scenarios and findings facilitated by our design and implementation.

1. Introduction

The structure of our outgoing emails (i.e. the number of people we write to, the frequency of these emails, and their grouping into threads) may be an important indicator of our roles in an organization. The work of Ducheneaut and Bellotti [DB01] studied factors influencing the use of email in organizations and discovered that email use varies with individual roles. The persistence and ubiquity of email has important consequences for the management of information within organizations [PSO06].

In this paper, we focus on the abstraction and visual presentation of an individual's outgoing email records. In

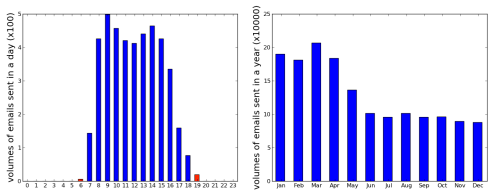

Figure 2: *Plots summarizing the number of emails sent in a day (left) and in a year (right) within the synthetic organization.*

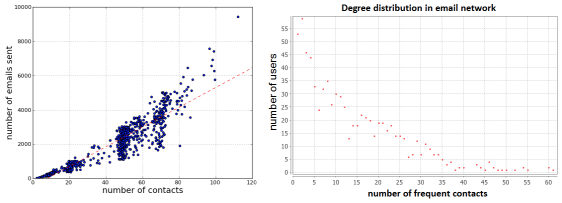

Figure 3: *Left: scatter plot showing the numbers of contacts and emails sent for all users. Right: distribution over the number of frequent contacts in the experimental dataset.*

[ARH12], the term 'abstraction' refers to a reduction in data complexity. Since a user typically makes contact with different groups of people by emails, we propose to summarize a user's outbound emails into different threads based on their recipients in abstracting email data. As a result, each thread has a starting time, a duration, and is essentially a collection of emails from the user to a group of participants. As different groups of participants may have overlaps, this abstraction produces, for each user, a bipartite graph between threads and participants.

For each user the proposed abstraction process reduces the data complexity from thousands of email records to dozens of threads that describe the user's active participation in key activities. We address the challenges of visualizing the abstracted data. In particular, we focus on (1) designing a glyph that offers an overview of a user's email activities while deriving and encoding attributes (i.e. activeness, membership) for participants and threads; (2) devising algorithms that improve upon the glyph design and drawing (i.e. giving details-on-demand using interactively controlled distortion, improving the visibility of each visual object in the layout).

Another challenge is to integrate the abstraction and visualization processes, providing a user interface for the analyst to browse, search and filter within a set of imported users. The goal is to show that, in combining the visualization with these basic functionalities, exploratory tasks like differentiating activity patterns and making associations are facilitated.

To address the challenges discussed above, we experimented with a dataset of 19.6 millions email records for 988 users in a synthetic organization, and built a software prototype to process and visualize this experimental data. Each record contains the sender's user ID, receiver's email address(es), a time-stamp of its delivery, and the number of attachments. Though users and email records are simulated, they reflect real-world email activity statistics including diurnal fluctuations and seasonal changes (Fig. 2), as well as relations and distributions among numbers of email records, contacts (Fig. 3). Our work makes the following contributions:

- **Data abstraction.** As a preprocessing step, email records are grouped per users. Then the abstraction step consists of *bundling* a set of email records into threads and building the thread-participant bipartite graph. We present bundling methods based on participant similarity score calculation.
- **Visualization design.** We have designed a glyph to visualize the abstracted user email activities. The glyph (Fig. 1) consists of a circular timeline to show the threads, a disk area to lay out the participants, and a set of thread lines to display email records with attachments. We provide rationale for the design, and present methods of encoding derived attributes such as participant activeness and membership information.
- **Layout and distortion techniques.** We devise interactive techniques that refine the visualization design. This paper presents two general techniques: (1) a force-directed method for placing circular nodes without overlapping, and (2) an angular fisheye distortion that gives details-on-demand on a circular timeline.
- **Results and analysis.** We present experimental results and analysis with synthetic organizational email records. Specifically we focus on: (1) How does the visual design help with identifying analysis objects? (2) How to combine the visualization with functionalities implemented in *ActivityVis* for exporatory tasks, and what are the consequent discoveries resulting from using it?

The rest of this paper is organized as follows. Section 2 explains the research tasks concluded from the analysts' requirements and interests. Section 3 presents an expository account of related work. We present the details of abstraction and visualization processes and algorithms for force-directed packing and angular fisheye distortion in Section 4. Finally, Section 6 and Section 7 discuss and conclude our work.

2. Research Tasks

The work is motivated by the need of abstracting large quantities of email records and provide interactive visualization for microscopic analysis. As discussed by van Wijk [vW06], we bridge gaps from the work of designers by learning about the requirements from analysts. In this paper, we refer to the people involved in email activities as *users*, the recipients of

a user's outgoing emails as *participants*, and the people who make use of our implementation as *analysts*. We followed iterations of *nested model for visualization design and validation* [Mun09] in reaching the following research tasks at multiple levels.

T1: Show email activities.
The email records we have from the analysts are synthesized based on real-world meta-data email records captured by traffic sniffing without the text content. Each record has a time-stamp of its delivery and attributes showing the sender, recipients, and the number of attachments. By interacting with ActivityVis, analysts should be able to grasp the time-oriented information and the attribute values of the email records.

T2: Abstract and visualize based on users.
The data abstraction process should be based on and separated by users within the organization. One reason behind this requirement is that analysts have different permissions which eventually manifest in the access rights to sets of email records based on users. At a basic level, the users can rightfully retrieve their own email records and perform a self-analysis. Therefore, the visualization should be based on users as well. This consideration leads to the design that each visual glyph is dedicated to show one user's email activities.

In addition, if analysts can access the email records of multiple users, they should be able to navigate through users and select one for visualization. It is better if the analyst can make the next user selection based on the current visualization. Also, they might want to choose multiple users for a comparison for activity pattern discovery.

T3: Encode derived attributes.
The visual glyph should be able to encode derived attributes such as the activeness of participants and their memberships which may reveal anomalies. There are many ways to define anomalies and activeness. Based on discussions with our clients, the visualization is designed to show: (1) Attachments sent at unusual times (between 10pm and 6am) during a work day, (2) Participants outside of the organizational domain, (3) Numbers of participants outside the organization in a thread, (4) Numbers of threads that a participant get involved in. As shown in Fig. 1, the objects in (1) or (2) can be encoded by binary colors. The number in (3) is mapped to a ratio (between outsider and the total numbers of participants) and encoded by a color scale. To show participant activeness the number in (4) is then encoded using size and position.

T4: Improve visibility.
Most importantly, the layout should preserve the visibility of each visual object. In drawing a thread-participant bipartite graph using the encoding described in T3, we observe that visual objects may occlude each other (e.g. Fig. 7). To solve the overlapping problem, we devise geometric techniques, constraint packing and angular fisheye distortion, to reduce the visual occlusion on the circular timeline or in the layout inside. Technical details and results are presented in Sec. 4.3 and Sec. 4.4.

3. Related Work

For the requirements described in Sec. 2, we abstract email archive data based on individual people. In the past, different types of abstraction have been proposed for analyzing email archives. For example, Li et al. [LHS04] identify and visualize individual email message flows and cliques of organizational email accounts for detecting anomaly based on clique violation. Perer et al. [PSO06] summarize and compare the trends of email activities based on the numbers of messages over the duration of an e-mail archive for relationship discovery. Viégas et al. [VGD06] presents a visualization showing detailed email exchange history between the user and friends for relationship characterization. Abstracting time-series data into meaningful levels is practiced among works from other domains, such as medical analysis [WPQ*08] [WGGP*11] and power consumption [HMJ*11] [HMJ*11]. If privacy is preserved, mining the actual message contents of personal emails leads to standard client-side applications, such as TIARA [WLS*10] and MUSE [HLH11], which extract cues (keywords, contact groups) displayable by stacked graph visualization. Finally, as reported in a controlled experiment by Aigner et al. [ARH12], encoding the abstracted data by composite visual representation can make the interpretation faster and even more reliable.

Pertaining to the visual design, circular shapes have been used for presenting events, such as geological and political data, that are evolving and periodic in nature (Fig. 4). Similar designs have been found in different application domains such as security analysis [Erb03] [ZN12], traffic visualization [FFM12], power consumption [JSMK13]. In these designs, time-series data are usually laid out on a circular timeline of clockwise orientation. One reason of using circles, as mentioned in [FFM12] and [ZN12] is that circular shapes are perceived as separate items pre-attentively and they facilitate viewing an entire time series as an entity. We share the same rationale by using a circular timeline to layout the activity threads for each user. Also, each thread is itself a timeline orthogonal to the circle. This allows the thread line to gain more space as it extends radially outward. We provide a cursory glance at related work addressing graphical fisheye distortion, force-directed packing and relation analysis techniques. In the survey on distortion-oriented techniques by Leung et al. [LA94], graphical fisheye transformation has been applied to a one-dimensional axis, Cartesian or polar space. Here we provide the application of fisheye transformation to angular space, for showing details on a circular timeline. The force-directed technique used in our work is similar to that in the streaming bubble chart presented by Huron et al [HVF13]. They use a force-directed

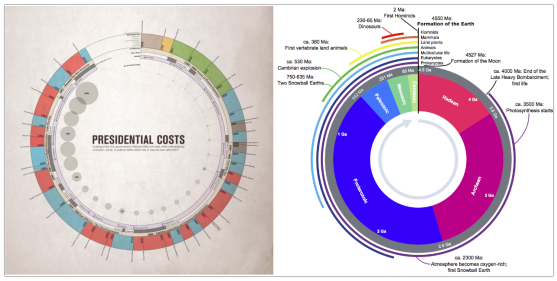

Figure 4: *Circular shapes appeared in designs for presenting time-oriented political data [Mer10] (left) and geological events [WH10] (right).*

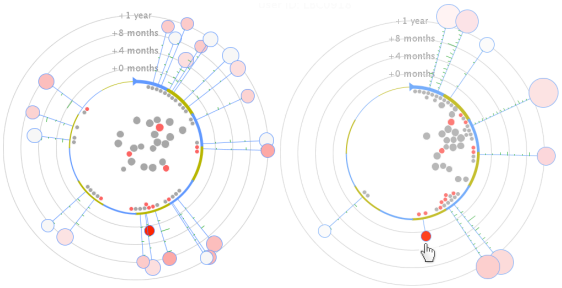

Figure 5: *Left: there are 23 threads after two iterations of merge. Right: 9 are left after another two iterations. Notice that there is a (short) thread not being merged with any other.*

method with collision detection to guide the movement of circular nodes without overlapping during live updates to an area chart. Incoming data are typically shown as circular nodes, which decrease in size as they join the aggregated area in the chart, to simulate the visual process of sedimentation Xu et al. [XDC*13] present a framework for visually analyzing both social and the corresponding item network, demonstrating the overlapping interest among people in the social network.

4. Abstraction and Visualization

In this section we describe in detail how ActivityVis transforms a collection of email records into visual illustrations, as shown in Fig. 1.

4.1. Definition and Handling of Data Entities

Notations. Let R denote the collection of email records with userID as the primary key. As a preprocessing step, we sort R according to the sender's userID: R is split into sets of records where each set R_U is associated with a user U. Given R_U, we further bundle the records in it into a list of threads so that each record belongs to one thread. Assume that T_U is the list of threads for U, and P_U is the group of people contacted by U. For each thread t in T_U, let $t.start$ and $t.end$ represent the start and end times of the thread, and let P_t represent the set of people who participated in t. Similarly for each participant p in P_U, let T_p represent the set of threads that involve p. Using $E_U = \{(p,t)|p \in P_t, t \in T_U\}$ to denote the set of thread-participant relations for user U, the email activities of a user can be abstracted to a bipartite graph $G_U = (T_U, P_U, E_U)$. Next we discuss details of the abstraction process which bundles the set of records R_U into a list of threads T_U and builds the bipartite graph G_U.

Bundling threads.
To bundle a set of records R_U into a list of threads T_U, we consider two criteria based on *participant similarity* and *temporal locality*. First, we initialize each record $r \in R_U$ as a thread. Next we use a greedy approach [SCL*12] to find pairs of *mergeable* threads with the highest similarity scores. Then we merge the idetified pairs and update the thread list. Mathematically, two mergeable threads ti and tj should satisfy:

$$PeriodBetween(ti.start, tj.end) < one\_year. \quad (1)$$

Since the time from $ti.start$ to $tj.start$ is less than or equal to that from $ti.start$ to $tj.end$, the period between two mergable threads is constrained by Inequ. 1, which also limits the duration of any merged thread. In addition, the similarity score of ti and tj should be above a certain threshhold in order for them to be considered mergable. The similarity score between ti and tj is computed based on the overlap of their participant groups:

$$similarity(ti, tj) = \frac{|P_{ti} \bigcap P_{tj}|}{|P_{ti}| + |P_{tj}|}. \quad (2)$$

It is defined by dividing the number of shared participants, $|P_{ti} \bigcap P_{tj}|$, of two threads by the sum of their participant group sizes $|P_{ti}| + |P_{tj}|$. In this way, two mergeable threads must share many participants as they grow in sizes.

The bundling is an iterative process. The iteration stops if there are no pairs of mergable threads or if it reached the maximum number of iterations. Usually each merge iteration leads to a decrease in the number of threads, as shown in Fig. 5; however, this process preserves a thread if its participants do not overlap participants of other threads.

The thread-participant bipartite graph is automatically built from the list of threads T_U with their participant groups $\{P_t, t \in T_U\}$. Specifically, the group of contacts P_U of user U is the union of all participant groups of threads: $P_U = \bigcup_t P_t$. The list of threads T_p involving a participant p is constructed through its definition: $T_p = \{t, p \in P_t\}$.

In our implementation, the maximum number of iterations is set to 5 in order to reduce the number of threads to a suitable range (<100) for most of the users in experimental dataset. For each user, the abstraction process takes a variable time, from 3 to 580 miliseconds, mainly depending on the number of email records.

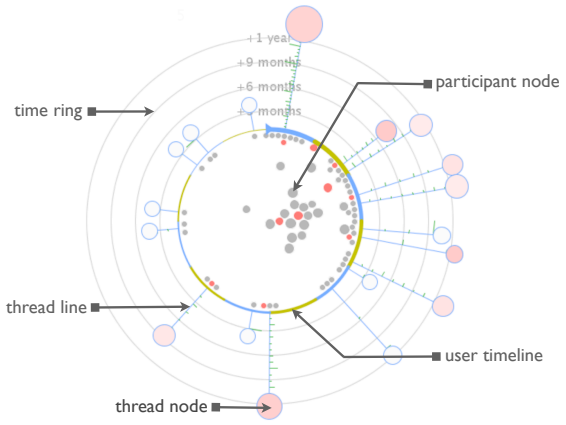

Figure 6: *Basic design and meanings of the visual objects: The User timeline is composed of 12 arcs, each denotes a period of one month. For a thread, the number of its participants and their memberships are reflected by the size and color of the thread node. Its duration is mapped to the line's length, which is further labeled by time rings. For a participant, activeness is encoded by both position and size, and membership is indicated by the color of the participant node.*

4.2. Basic Design of Visual Objects

This section describes visual glyph design and encoding, as illustrated by Fig. 6. The period of time represented by the timeline is predetermined to be one year starting from the earliest timestamp identified from the records. In addition to the circular timeline, we also have a list of thread lines orthogonal to the circular timeline: Each thread is represented by a straight line that extends radially outward with its position on the circular timeline determined by its starting time. The numbers of attachments in the email records of a thread are plotted as tangential bars on the thread line.

To help the analyst infer the duration of a thread, we use time rings, which are concentric circles with increasing radii, to label the thread lines. We further attach a node at the end of a thread line, for encoding two derived properties: the thread node's radius encodes the participant group size; the node color encodes the proportion of oursiders in the group. Also, attaching a node makes it easy for the thread to be interactively selected by analysts. Here, the advantage of using a circular timeline manifests in allowing more screen space for a thread node as the thread line extends radially outward.

The group of participants for a user are represented by nodes of variable size and color inside the user's circular timeline. We next present details about their layout and encoding.

4.3. Node Layout

Many visualization tasks require solving the layout of graph nodes in a convex region as described by the classical problem of barycentric layout [Tut63]. In our case with thread-participant bipartite graph, the problem is greatly simplified by a design where threads are laid out on a circular timeline.

Inside the circular timeline, we lay out participant nodes while considering the following criteria:

- Spatial affinity. It is better for a group of participant nodes sharing the same thread close to each other, and also close to the thread's position on the circular timeline, in order to infer clusters of participants from the layout.
- Activeness encoding. A participant involved in many threads over the period of time is considered as an active participant. The corresponding node should be emphasized in a way that attracts visual attention.
- Visibility. The group of participant nodes in the layout may overlap with each other. In order to make each node visible for selection and counting, overlappings in the layout should be minimized.

After investigating various design alternatives, we chose a solution that works in two steps. First we initialize participant node positions by weighted combinations of threads' positions on the circular timeline. Mathematically, let C_t represent the contacting point of thread t with the circular timeline. The position X_p of node p is set to,

$$X_p = \Sigma_{t \in T_p} \frac{1}{|T_p|} C_t, \tag{3}$$

From the above equation, X_p is determined by positions of the threads involving p. Therefore, nodes associated with similar sets of threads are close to each other. This produces a layout in which active participants (ones associated with many threads over the period of time denoted by the circular timeline) tend to be placed at the center. Less active participants such as those associated with only one thread are placed adjacent to the circular timeline, near the threads to which they belong.

It is possible that a participant involved in two threads (with 6 months difference in their starting times) is placed at the center as well. To differentiate this with other more active ones, we also map the number $|T_p|$ to node p's radius r_p. Therefore, a participant's activeness is emphasized by both of its position and size (Fig. 6).

Packing nodes.
The weighted combination in the first step produces an initial layout in which the group of nodes with various radii overlap with one another, as shown in the left of Fig. 7. To minimize the overlapping, we use a force-directed method [HVF13] to iteratively move any intersecting pair of nodes apart while keeping them gathered and confined in the circular timeline in the second step. This resembles a constrained circle packing process which also applies to reducing the overlapping among thread nodes (Fig. 7). In particular, the details of each iteration are as follows,

Detect: Check each pair of nodes $p : (X_p, r_p)$ and $q : (X_q, r_q)$

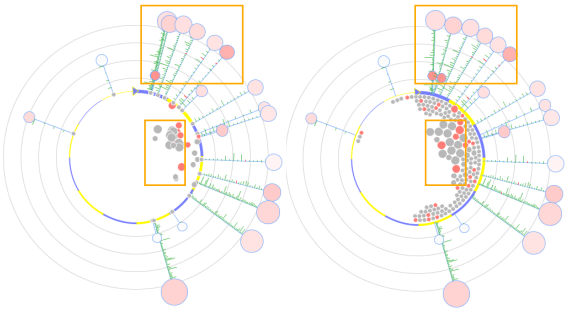

Figure 7: *In the left figure, the highlighted set of threads or participant nodes are occluded each other, losing their visibility for selection. During force-directed packing, overlapping nodes repel each other, while remaining close to their original positions and confirmed to constraints. The result is shown on the right.*

for intersection. Put (p,q) into a list L if they overlap each other.

Repel: Compute an offset vector $V_{pq} = \frac{r_p+r_q-|X_pX_q|}{2|X_pX_q|}\vec{X}_pX_q$ for each pair of nodes in L. Move X_p along V_{pq} and X_q along $-V_{pq}$. Empty list L on finishing.

Adjust: Test each node if it is contained in the circular timeline and move it towards the center if not.

In our implementation, the number of iterations for packing is 10, which enables making every node visible for most of the users with $|P_U| < 100$ in the experimental dataset. For each user, the rendering time varies from 8 to 43 milisecconds including the layout process. Compare to the data processing time in Sec. 4.1, we report that the rendering time is generally less and more uniformly distributed.

4.4. Interaction

We have presented a compact design for visualizing a user's email activity. Though it is possible that additional information can be encoded by the design, it becomes difficult for all details to be properly inferred from the static visualization. To help analysts uncover some important details, we enhance the visualization by providing two interactions, together with a complementary user interface.

Angular fisheye distortion.
Recall that the contacting point C_t of thread t with the circular timeline is $C_t = O + (r\sin\alpha_t, r\cos\alpha_t)\}$ where O is the center, r the radius. We focus on α_t, that is linearly mapped to $[0,2\pi]$ according to the starting time of t. The problem is that if several threads start in a short period of time, then contact points of these thread lines are very close to each other, making it hard to tell the order of their occurrence, as shown in the left of Fig. 8. In order to reveal the order information in this case, we provide interactive fisheye distortion to angles of thread lines, and to those of the circular timeline as well.

Formally, let α_m represent the relative angle from the line segment connecting the mouse position and the center. When fisheye distortion is on, a angle α_0 is transformed to α_1 based on its normalized difference δ from α_m:

$$\delta = \frac{\alpha_0 - \alpha_m}{\pi}(\pm 2), \tag{4}$$

where (± 2) is to let δ within the range $[-1,1]$, over which the standard fisheye distortion function, $T(\delta)$, is defined [FK96]:

$$T(\delta) = \frac{(F_P + F_D)\delta}{F_P + F_D|\delta|}, \tag{5}$$

where $\frac{F_D}{F_P}$ is the amplification factor, which is 2.0 in our implementation. Then distorted angle α_1 is computed,

$$\alpha_1 = \alpha_m + T(\delta)\pi \tag{6}$$

In Fig. 8 (center, right), we show the results after applying fisheye distortion to two regions that originally contain multiple threads happening in a very short period of time. 12 timeline arcs are filled with alternating colors to reveal how angles are transformed everywhere on the circular timeline. Though distorted, the threads' order on the timeline become easy to see near the mouse.

Highlighting.
Although visibility is improved by packing participants in the layout, it may be difficult to tell which nodes belong to a particular thread, or which threads involve a certain participant. We implement a selection-based highlighting mechamism to address this issue: If the cursor hovers over a node p, both p and the threads in T_p are highlighted, as shown in Fig. 12. Similarly, if a thread t is selected, both t and P_t are highlighted, as shown in Fig. 13.

User interface.
In ActivityVis, a GUI (Fig. 9) is provided to browse, search and filter based on abstracted user data. This component originally serves as a portal for creating different views: Analysts can select a user and create a main view for a visual investigation of the user's activity. Or, they may select several users and create multiple views for a comparison as illustrated in Fig. 11. When comparing multiple users, we only create simplified views with details and interactions suppressed.

From discussions with domain analysts, we conclude that it is better if the analyst can make the next user selection based on the current visualization, as specified by the second task requirement in Sec. 2. Hence we add functionalities to make the visualization and the GUI components interact with each other, as shown in Fig. 12: The three steps describe a task of identifying another user contacted by the current user being rendered in visualization and further investigating how the current user plays an role in the newly identified user's email

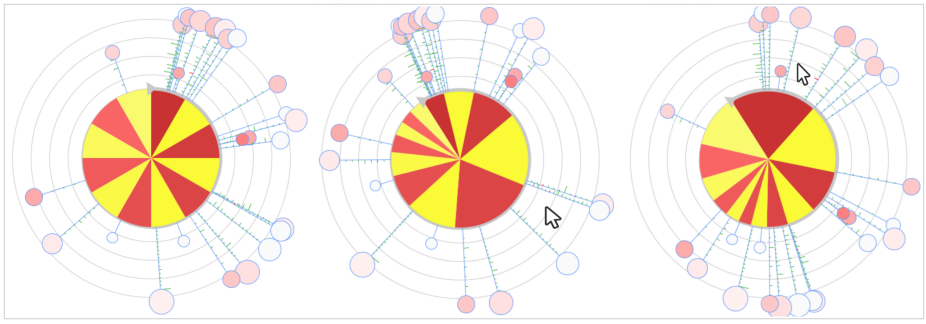

Figure 8: *When angular fisheye distortion is on, the angle difference (the span between any two points on the circular timeline) can either expand on the side same as the cursor or compress on the other side.*

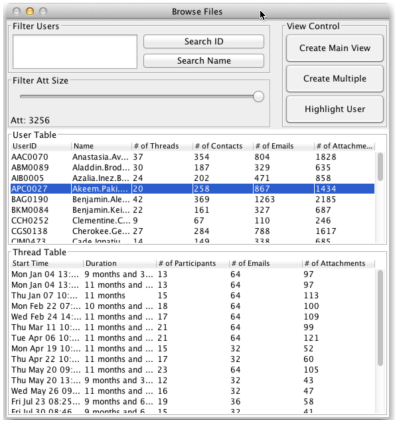

Figure 9: *The GUI mainly consists of a user table and a thread table for listing out a collection of users and the set of threads for the selected user. It also has a table control panel for filtering and searching among the collection, and a view control panel for generating different views.*

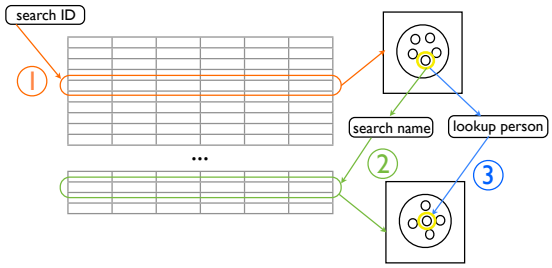

Figure 10: *Three steps of finding a association: (1) The analyst first searches with id within a collection of users and brings up user A's records to generate a visualization of A's email activity. (2) By looking and interacting with the view, the analyst may identify an active participant B and record his email address. (3) The analyst then searches with B's name, and generate a visualization of the email activity of B if B's records are in the collection. Moreover, the analyst can highlight A in B's view. A corresponding case scenario is given in Section 5 and Fig. 12.*

activities. A case scenario is given in Fig. 12 and Section 5 presents the analysis.

5. Results and Analysis

This section augments previously discussed design and techniques with concrete examples.

Experiment setup.
We use a dataset of 19.6 million email records of 988 users in a synthesized organizational social network. As discussed in Sec. 1, these users and email records are generated according to real-world email activity statistics. The 998 sets of records are divided into 26 batches according to the initial letter of each user's surename. The analyst can load one or several batches into ActivityVis, which first automatically summarizes email records into threads and then displays a list of input users. All visualizations in ActivityVis are implemented in Java with the Processing runtime library. Next we analyze different query types of analysts' interests, and findings assisted by our implementation.

Direct queries.
Direct queries involve fact-finding tasks such as telling if an email with attachments is sent during an unusual time, or whether a participant belongs to a particular thread. In ActivityVis, the answers to direct queries are straightforwardly shown by table or in the visualization. For example, in Fig. 13, there is an email record with a large number of attachments sent during an unusual time (indicated by a long red bar). It belongs to a thread of nine participants, two of whom are outsiders (denoted by the two red nodes), and another two who are very active (shown as the two relatively large gray nodes close to the center).

Indirect queries and findings.

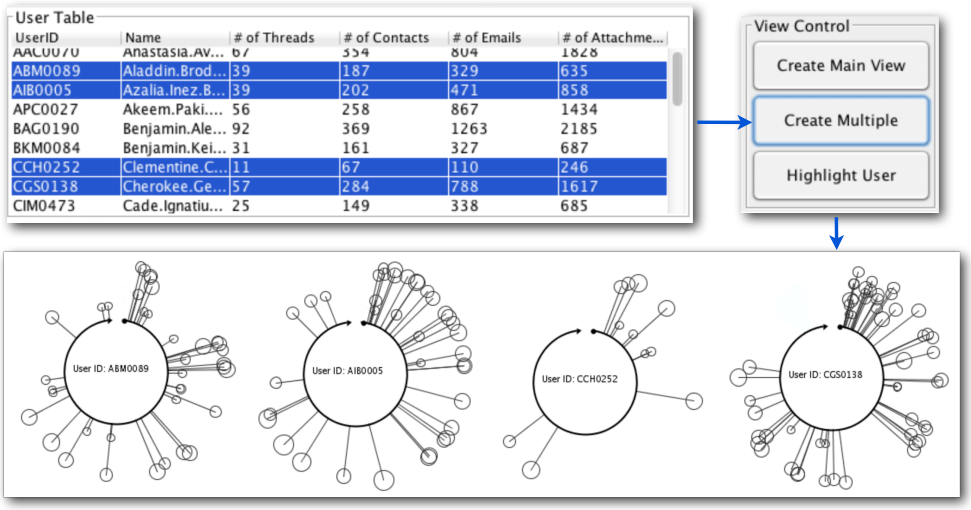

Figure 11: *Four users are selected and simplified views of their activities are shown side-by-side for a visual comparison: Though having the same number of threads, most last longer for the second user compared to the first user. Though having very different numbers of threads, the overall distribution on the circular timeline in the third view appears similar to the fourth.*

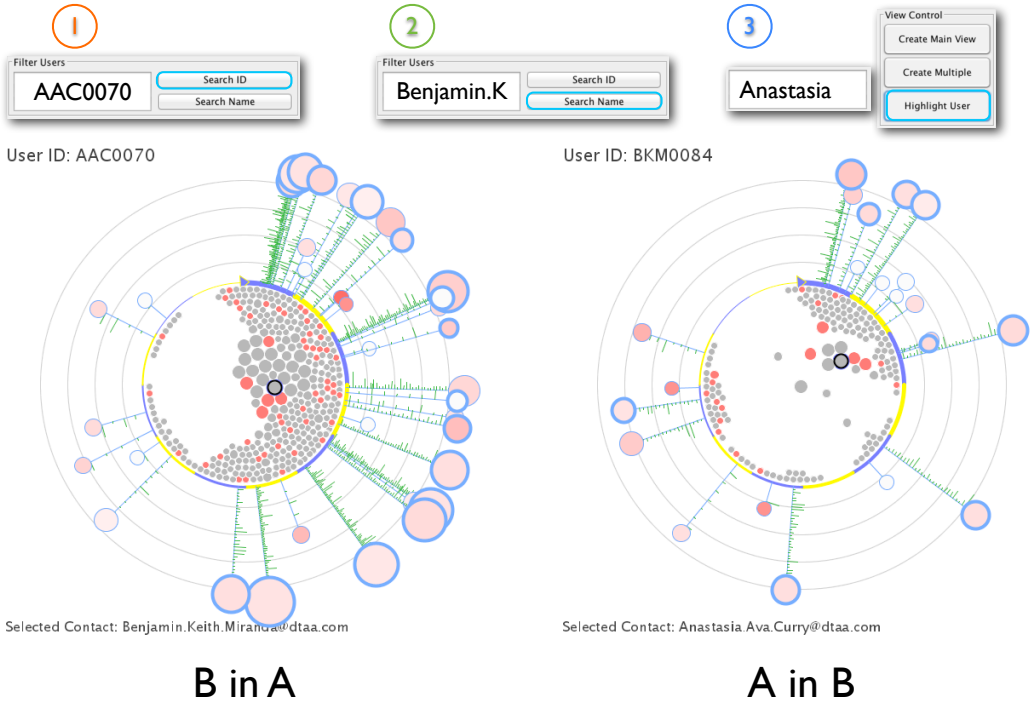

Figure 12: *A case scenario of associating a user with another: We first search with ID AAC0070, and bring up the visualization of A's email activity. We identify an active contact B, Benjamin.Keith.Miranda@dtaa.com, who participates in many threads as highlighted in the visualization. Then we search with B's name and bring up the visualization of B's activity (B is in the collect as well.) Moreover, we highlight user A together with the threads involving A in B's visualization. The resulting two views, "A in B" and "B in A", show how A and B play a role in each other's outbound email history.*

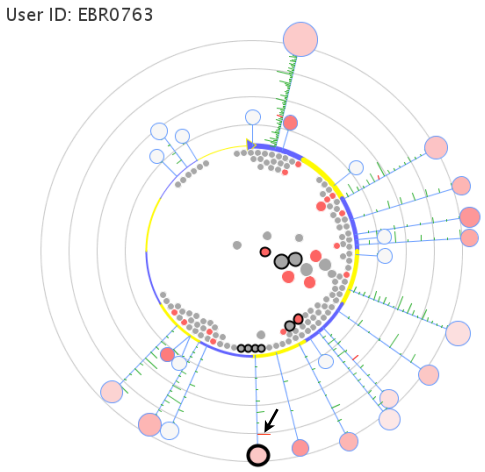

Figure 13: *The long, red bar on a thread line indicates an email record with a large number of attachments sent during an unusual time of day. Selecting the corresponding thread further highlights the group of 9 participants, including 2 outsiders, contacted by the user.*

Indirect queries suggest exploratory tasks, of which the answers are not absolute, typically involving analyst effort to interact with several components of ActivityVis. The focus is on the following two indirect questions: What are some of the behavior patterns that can be observed from a visual comparison? And how do a pair of users participate in each other's activities? We find results shown by Fig. 11 that answer the first question: Two users, *ABM*0089 and *AIB*0005, have the same number of threads. However, a thread of *AIB*0005 tends to last longer than a thread in *ABM*0089 since he has fewer 'short' thread. Another two users, *CCH*0252 and *CGS*0138, have very different numbers of threads. But their views reveal a similarity in the overall distribution. To answer the second question, we provide a case scenario in which the analyst makes a association between user *AAC*0070 and user *BKM*0084: the visualizations show that *B* actively participates in many threads of *A*, and so does *A* in *B*.

6. Discussion

Before designing the visualization, we performed a statistical study on the experimental dataset. We found this step useful as it gave the estimates of important 'dimensions' of the data, such as the typical number of emails or contacts for a user. This allowed us to devise the visualization techniques accordingly. For example, most users's numbers of contacts do not exceed 100 (Fig. 3). Therefore, we chose force-directed packing which is fast for laying out a group of less than a hundred nodes. Force-directed methods may be very slow for a large set of nodes as it is a N-body simulation which has quadratic complexity if an accelerating data structure is not employed. On the other hand, the typical number of emails sent by a user over one year is above 3000, which would be overwhelming to show if they are not summarized and bundled into threads.

This work is a collaboration between computer graphics designers and data analysts. The primary target user of ActivityVis is an analyst, who explores the email activity patterns of individuals in the organization. We provide in this paper the task requirements discussed during the collaboration, and plan to do controlled user study as future work.

7. Conclusion

This work is motivated by the tremendous opportunity in microscopic analysis of email activities within an organization, in which a user typically makes contact with different groups of people by email. We have presented a tool to abstract user email records into thread-participant bipartite graphs, and to visualize the abstracted data.

We designed a hierachical glyph for representing the thread-participant bipartite graph where threads are laid out on a circular timeline and the group of participants are placed inside. To improve the visual design, we devised general techniques: force-directed node packing and angular fisheye distortion for improving the visibility or showing details-on-demand. We experimented with datasets that simulate real-world email activities in an organizational network. The case scenarios show that our design and implementation facilitate differentiating activity patterns, or making associations among users.

References

[ARH12] AIGNER W., RIND A., HOFFMANN S.: Comparative evaluation of an interactive time-series visualization that combines quantitative data with qualitative abstractions. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 995–1004. 2, 3

[DB01] DUCHENEAUT N., BELLOTTI V.: E-mail as habitat: an exploration of embedded personal information management. *interactions 8*, 5 (2001), 30–38. 1

[Erb03] ERBACHER R. F.: Intrusion behavior detection through visualization. In *Systems, Man and Cybernetics, 2003. IEEE International Conference on* (2003), vol. 3, IEEE, pp. 2507–2513. 3

[FFM12] FISCHER F., FUCHS J., MANSMANN F.: Clockmap: Enhancing circular treemaps with temporal glyphs for time-series data. *Proc. EuroVis Short Papers, Eurographics* (2012), 97–101. 3

[FK96] FORMELLA A., KELLER J.: Generalized fisheye views of graphs. In *Graph Drawing* (1996), Springer, pp. 242–253. 6

[HLH11] HANGAL S., LAM M. S., HEER J.: Muse: Reviving memories using email archives. In *Proceedings of the 24th annual ACM symposium on User interface software and technology* (2011), ACM, pp. 75–84. 3

[HMJ*11] HAO M., MARWAH M., JANETZKO H., SHARMA R., KEIM D., DAYAL U., PATNAIK D., RAMAKRISHNAN N.: Visualizing frequent patterns in large multivariate time series. In *IS&T/SPIE Electronic Imaging* (2011), International Society for Optics and Photonics, pp. 78680J–78680J. 3

[HVF13] HURON S., VUILLEMOT R., FEKETE J.-D.: Visual sedimentation. *Visualization and Computer Graphics, IEEE Transactions on 19*, 12 (2013), 2446–2455. 3, 5

[JSMK13] JANETZKO H., STOFFEL F., MITTELSTÄDT S., KEIM D. A.: Anomaly detection for visual analytics of power consumption data. *Computers & Graphics* (2013). 3

[LA94] LEUNG Y. K., APPERLEY M. D.: A review and taxonomy of distortion-oriented presentation techniques. *ACM Transactions on Computer-Human Interaction (TOCHI) 1*, 2 (1994), 126–160. 3

[LHS04] LI W.-J., HERSHKOP S., STOLFO S. J.: Email archive analysis through graphical visualization. In *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security* (2004), ACM, pp. 128–132. 3

[Mer10] MERCER R.: History of the united states in a circle. http://flowingdata.com/2010/06/13/history-of-the-united-states-in-a-circle/, 2010. 4

[Mun09] MUNZNER T.: A nested model for visualization design and validation. *Visualization and Computer Graphics, IEEE Transactions on 15*, 6 (2009), 921–928. 3

[PSO06] PERER A., SHNEIDERMAN B., OARD D. W.: Using rhythms of relationships to understand e-mail archives. *Journal of the American Society for Information Science and Technology 57*, 14 (2006), 1936–1948. 1, 3

[SCL*12] SHI C., CUI W., LIU S., XU P., CHEN W., QU H.: Rankexplorer: Visualization of ranking changes in large time series data. *Visualization and Computer Graphics, IEEE Transactions on 18*, 12 (2012), 2669–2678. 4

[Tut63] TUTTE W. T.: How to draw a graph. *Proc. London Math. Soc 13*, 3 (1963), 743–768. 5

[VGD06] VIÉGAS F. B., GOLDER S., DONATH J.: Visualizing email content: portraying relationships from conversational histories. In *Proceedings of the SIGCHI conference on Human Factors in computing systems* (2006), ACM, pp. 979–988. 3

[vW06] VAN WIJK J. J.: Bridging the gaps. *Computer Graphics and Applications, IEEE 26*, 6 (2006), 6–9. 2

[WGGP*11] WONGSUPHASAWAT K., GUERRA GÓMEZ J. A., PLAISANT C., WANG T. D., TAIEB-MAIMON M., SHNEIDERMAN B.: Lifeflow: visualizing an overview of event sequences. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2011), ACM, pp. 1747–1756. 3

[WH10] WOUDLOPER, HARDWIGG: Geologic time scale. http://en.wikipedia.org/wiki/Geologic_time_scale, 2010. 4

[WLS*10] WEI F., LIU S., SONG Y., PAN S., ZHOU M. X., QIAN W., SHI L., TAN L., ZHANG Q.: Tiara: a visual exploratory text analytic system. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* (2010), ACM, pp. 153–162. 3

[WPQ*08] WANG T. D., PLAISANT C., QUINN A. J., STANCHAK R., MURPHY S., SHNEIDERMAN B.: Aligning temporal data by sentinel events: discovering patterns in electronic health records. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (2008), ACM, pp. 457–466. 3

[XDC*13] XU P., DU F., CAO N., SHI C., ZHOU H., QU H.: Visual analysis of set relations in a graph. In *Computer Graphics Forum* (2013), vol. 32, Wiley Online Library, pp. 61–70. 4

[ZN12] ZHUO W., NADJIN Y.: Malwarevis: entity-based visualization of malware network traces. In *Proceedings of the Ninth International Symposium on Visualization for Cyber Security* (2012), ACM, pp. 41–47. 3