國立臺灣大學電機資訊學院資訊工程學系

碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

電子郵件資料中事件演進模式與人際互動關係之視覺化研究

EmailMap: Visualizing Event Evolution and Contact Interaction within Email Archives

黃莉婷

Li-Ting Huang

指導教授：陳炳宇 博士

Advisor: Bing-Yu Chen, Ph.D.

中華民國 101 年 7 月

July, 2012

致謝

感謝陳炳宇教授給我很大的空間，去探索喜歡的研究方向。

感謝莊永裕老師、楊傳凱老師、王昱舜老師、沈漢威老師給我的許多寶貴建議。

感謝阿山一路的帶領，告訴我研究的過程比結果更重要、研究要做的開心。

感謝吳學侑，感謝你，過去、現在，和未來，像我的半個家人。

感謝小鐵，當我的好鄰居好朋友聽我講些有的沒的。

感謝Calvin、Xavier、楊立德在我兩頭燒忙不過來時的體諒和幫忙。

感謝兩百、漾漾、小莊、桃爸、亮岑、大叔，這兩年一起努力的同學，也感謝你們給我建議、提出想法，點出了許多我原本沒有想到的面向。

感謝毛，投 InfoVis 時在實驗室玩耍到三點活蹦亂跳的還幫買飲料，讓趕稿變得有趣。

感謝實驗室的學長姐、同學、學弟妹們，解答我的疑惑、陪伴我、聊天討論。

Jag skulle vilja visa min tacksamhet gentemot Henrik Larsson, som har stöttat mig och varit vid min sida genom vått och torrt.

感謝我的父母、哥哥，我永遠最堅強的後盾與支柱。

中文摘要

電子郵件資料中有大量關於人們如何和連絡人互動，以及生活中的事件如何演進、發展的資訊。理解、組織這些資料可以幫助人們了解並且綜觀過去的生命經驗。儘管過去已有不少研究著力於電子郵件的視覺化，但大部分的研究主題仍集中在下列兩者之一：理解電子郵件中之事件發展、或人們和連絡人互動關係之變化，尚未有研究能提出完整的方式整合這兩項資訊。在本研究中，我們提出EmailMap – 一個將電子郵件視覺化的系統。在此系統中，我們整合事件發展、和連絡人互動關係的資訊至一個單一的系統介面，讓使用者能夠透過兩個互補的資訊，充分理解隱藏在大量電子郵件資料中的脈絡和資訊。我們設計了兩種視覺化元素來呈現這些資料：事件流（event flow）和聯絡人追蹤（contact tracks）：事件流以流線方式呈現過去事件之演變，幫助使用者理解資料較為全觀性的特徵與結構；聯絡人追蹤則用以呈現人們之間互動的情況。最後，我們透過質性使用者測試，驗證了EmailMap系統之有效性，說明此系統能幫助使用者整合過去事件之演變與人們互動關係之變化，提供更豐富的回想經驗。

Abstract

Email archives contain rich information about how we interacted with different contacts and how events evolved throughout time. Making sense of the archived messages can be a good way to understand how things evolved and progressed in the past. Although much work has been devoted to email visualization, most work has focused on presenting one of the two aspects of email archives: discovering the evolution of emails and events, or the relationship between contacts and ego over time. Very few systems support an integration of both. In this paper, we present EmailMap, an email visualization which integrates the information of both events and contacts into a single view, enabling users to make sense of their email archives with complementary contextual information. Two visualization components are designed to portray complex information within the email archives: event flow and contact tracks. The event flow illustrates the evolution of past events, helping the users to grasp high-level pictures and patterns of their email archives. The contact tracks reveal the interaction between ego and the contacts. We also conducted a qualitative user study to demonstrate the effectiveness and usefulness of EmailMap in helping users to integrate past life events with the interaction between people into a rich and meaningful reminiscence.

Contents

致謝 **i**

中文摘要 **ii**

Abstract **iii**

1 Introduction **1**
1.1 Motivation 1
1.2 Problem Statement 2
1.3 Proposed Method 2
1.4 Organization 3

2 Related Work **4**
2.1 Email Visualization 4
2.2 Timeline-based Visualization 6
2.3 Flow Map Layout 7

3 EmailMap Design **8**
3.1 Dual Design Focus 8
3.2 Events as Flow 9
3.3 Contacts as Tracks 11
3.4 Interaction 14
3.4.1 Conversation Thread Interaction 14
3.4.2 Contact Interaction 14
3.4.3 Smart Zooming 15
3.4.4 Hovering and Double-Clicking 15

4 Email Processing **17**
4.1 Constructing Hierarchical Email Structure 17
4.1.1 Similarity Method 17
4.1.2 Hierarchical Email Clustering 19
4.1.3 Hierarchical Email Structure 19
4.1.4 Connecting to the Flow Root 21
4.2 Keyword Extraction 21
4.3 Contact Processing 21

5 Computing Layout **23**
5.1 Computing Event Flow Layout 23
5.1.1 Input Hierarchical Email Structure 23

5.1.2 Objective Function 25
5.1.3 Optimization 31
5.2 Computing Contact Tracks Layout 32

6 Results and Discussion 38
6.1 Case Study 38
6.1.1 Case 1: EmailMap Project 38
6.1.2 Case 2: A Trip to Norway 40
6.1.3 Case 3: Organizing PacificVis 41
6.2 Evaluation 41
6.2.1 Methodology 41
6.2.2 Results 44

7 Limitation and Future Work 47
7.1 Limitation 47
7.2 Future Work 48

8 Conclusion 50

List of Figures

3.1 An example of an event flow, which consists of three visual elements: branch, email, and thread. 10

3.2 The contacts are represented as curved tracks. When two tracks intersect, the sender is encoded as a solid circle of his/her corresponding color key with white border line, and the receivers are encoded by concentric rings of their corresponding color keys. 11

3.3 Facebook's notification as an example of an active contact. 12

3.4 A manager as an example of an active contact. 12

3.5 A supervisor as an example of a passive contact. 13

3.6 A group of students as examples of passive contacts. 13

3.7 Two people working closely on EmailVis project as examples of balanced contacts. 13

3.8 Two friends exchange emails from time to time as examples of balanced contacts. 14

3.9 Three different sizes of email nodes depicting three different numbers of email message receivers. 15

4.1 The three clustering conditions, (a) temporally overlap, (b) non-overlap, temporally close to each other, and (c) non-overlap, temporally far away from each other. 20

5.1 Displaying an example of the input structure for the optimization: the root node is colored in red, the email nodes are colored in white, the extend nodes are colored in blue, and the structure node is colored in orange. In this figure, the positions of nodes are adjusted in order to clearly display all four types of nodes. 24

5.2 The results of optimizing the energy function Ω with and without the weight ${w_{\mathbf{v}'_{\mathbf{c}}}}^2$. Notice that the thickness of each branch indicates the value of $w_{\mathbf{v}'_{\mathbf{c}}}$. (a) The input tree structure. (b) The result with ${w_{\mathbf{v}'_{\mathbf{c}}}}^2$. (c) The result without ${w_{\mathbf{v}'_{\mathbf{c}}}}^2$. 25

5.3 The initial layout of a set of real data. Email nodes are colored in white. The structure node colored in orange is displayed in order to show the effect of the structure cost. 26

5.4 The optimization result of Fig. 5.3 by adopting the weights as follows: $w_s = 1, w_{str} = 0, w_o = 0$, and $w_t = 0$. 27

5.5 The optimization result of Fig. 5.3 by adopting the weights as follows: $w_s = 0.1, w_{str} = 10, w_o = 0$, and $w_t = 0$. 28

5.6 The optimization result of Fig. 5.3 by adopting the weights as follows: $w_s = 0, w_{str} = 0, w_o = 0.1$, and $w_t = 0$. 29

5.7 The optimization result of Fig. 5.3 by adopting the weights as follows: $w_s = 0, w_{str} = 0, w_o = 10$, and $w_t = 100$. 30

5.8 The optimization result of Fig. 5.3 by adopting the weights as follows: $w_s = 1, w_{str} = 1000, w_o = 500$, and $w_t = 1000$. 30

5.9 An input structure of a contact track. 34

5.10 The optimization result of Fig. 5.9 by adopting the weights as follows: $w_{ct\_s} = 100$ and $w_{ct\_d} = 0$. 35

5.11 The optimization result of Fig. 5.9 by adopting the weights as follows: $w_{ct\_s} = 0.1$ and $w_{ct\_d} = 100$. 36

6.1 The visualization of EmailMap project. Five different contact tracks are displayed. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1, w_{str} = 100, w_o = 500$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: w_{ct_s} = 1 and $w_{ct\_d} = 500$. 39

6.2 The visualization of EmailMap project. Two different contact tracks are displayed. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1, w_{str} = 100, w_o = 500$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: w_{ct_s} = 1 and $w_{ct\_d} = 500$. 39

6.3 The visualization of planning a trip to Norway. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1, w_{str} = 1000, w_o = 500$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: w_{ct_s} = 1 and $w_{ct\_d} = 100$. 40

6.4 The visualization of large email archives with over 2,215 messages. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1000, w_{str} = 100, w_o = 1000$, and $w_t = 1000$. 42

6.5 The visualization of large email archives with over 2,215 messages. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1000, w_{str} = 100, w_o = 1000$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: w_{ct_s} = 1 and $w_{ct\_d} = 500$. 43

6.6 The visualization of large email archives with over 2,215 messages. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1000, w_{str} = 100, w_o = 1000$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: w_{ct_s} = 1 and $w_{ct\_d} = 500$. 43

6.7 The visualization of large email archives with over 2,215 messages. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1000, w_{str} = 100, w_o = 1000$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: w_{ct_s} = 1 and $w_{ct\_d} = 500$. 44

7.1 The visualization of a master student's email. Six different contact tracks are displayed. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1, w_{str} = 100, w_o = 100$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: w_{ct_s} = 1 and $w_{ct\_d} = 500$. 48

List of Tables

Chapter 1

Introduction

1.1 Motivation

Email is one of the most popular applications in our daily life[1]. With many free email services offering increasing storage and decreasing cost, email users tend to save most of messages they received [?]. Therefore, it is very likely that most email users have large email repositories as time goes by. As a major communication tool, email plays a critical role in our daily activities and our interaction with other people. No matter it is a research team with overseas members working closely on a project, a company announcing new policies, an international coordinator drafting a new cooperating program, or a meeting organizer sending meeting appointments, it is not unusual to find email as the prominent tool in delivering the messages. In addition, email has evolved into a multi-purpose tool used for more than just sending messages [?] [?].

With its diversified usages, email has become a passive life-logging medium. Unlike keeping a blog or a diary, email has recorded our life without people adding additional efforts. Therefore, to understand email archives can be a good way to understand how things evolved and progressed in the past. However, to traverse thousands of emails can be tedious. It is also difficult to make sense of large amount of data. As present email clients do not offer an efficient solution, it is crucial to provide a tool that can facilitate making sense of the rich life stories lying in the mass email archives.

[1] http://www.radicati.com/?p=7261

1.2 Problem Statement

Our life (described in email archives) can be understood in two aspects: people and events. People relate to how we interact with our colleagues, friends, and families. Events represent what tasks we have been working on, and how different events evolved during the past. Most previous systems for handling email archives usually focused only on presenting one of the two aspects: the relationship of people (i.e., contacts) [?] [?] [?] [?] [?] [?] [?] [?] [?], or the events the email archives presented [?] [?] [?] [?] [?] [?] [?]. Visualizations focused on people portray the relationship between the contacts over time. On the other hand, visualizations focused on events display the relationship between the email threads, and how different email threads emerge and change over time. With these visualizations, users can only trace either the aspect of people, or the aspect of events, but not both of them.

However, these two aspects are often interwoven tightly in our life story. Our interaction with different people often relates to different events. For example, when collaborating on a project, we work closely with the project members. Moreover, when reminiscing about the past, the relationship of people and the relationship of events can offer complementary hints. For example, we can remember more details of an event by remembering how we interacted with the people involved, and vice versa. Thus, providing the integration of both aspects may be a better way to describe our life described in the email archives.

1.3 Proposed Method

In this paper, we present EmailMap, a visualization that helps users associate the relationship of people and the relationship of events. We purpose a new way to understand personal email archives by integrating the contact relationship and email relationship into a single view, in which one relationship is used as the context providing more information to the other. Specifically, emails are grouped into a set of hierarchical events, and illustrated as an event flow over the temporal coordinate. Related email messages can be browsed along the same branch of the flow. In addition, each contact is visualized as a smooth color-coded track that connects all the emails related to him/her. With the aid of this visualization, users are able to explore both the longitudinal relationship with their contacts as well as the relationship of emails/events at the same time, thus getting a more holistic understanding of the life stories lying in their email archives. To better understand how the visualization facilitates users' comprehension toward their email archives, we conducted a qualitative user study assessing the information it provides to users. Results show that EmailMap effectively helps users to reminisce the details of past cooperation and the evolution of life events. It also enables users to get an overall picture and patterns of their email archives that were unclear to them before. Specifically, our work presents two major contributions.

- **A visualization system with event flow and contact tracks** that facilitates the interactive exploration of email archives with an integral context information depicting the evolution of past life events and the interaction patterns between people.
- **An optimization-based layout algorithm** that computes smooth event flow out of large-scale email archive data.

1.4 Organization

The rest of the paper is organized as follows. Section 2 discusses related work and compares our work with previous email visualization methods. Section 3 presents our design choices and visual encodings. Section 4 states how we create a hierarchical email structure and how we get contact information through email data processing. Section 5 gives our layout algorithm which is based on an optimization technique. We state each objective function, show its effectiveness, and describe how we deal with large data. Section 6 presents and discusses results in three case studies and a small-scale qualitative user study. Section 7 discusses the limitation and some future research directions. In Section 8, we give a conclusion of this work.

Chapter 2

Related Work

2.1 Email Visualization

Several studies have explored visualizing email archives, however, they have mainly focused on either the relationship of people [?] [?] [?] [?] [?] [?] [?] [?], or the presentation of events/email threads [?] [?] [?] [?] [?] [?] [?]. Visualizations focused on people aimed to portray the relationship between people (i.e., contacts) over time. On the other hand, visualizations focused on events/emails aimed to display the evolution of events, or the relationship between emails.

Themail [?] showed the dyad relationship by visualizing keywords that characterize one correspondence with each contact the best. With the aid of text analysis, the content of email archives summarized the dyad relationship. By scanning the keywords, one can get a general idea of how the relationship has changed over time without going through the haystack. PostHistory [?], an ego-centric visualization of email archives, used a calendar panel and a contacts panel in its interface. The contacts panel showed the overall importance of the contacts. Once a contact was selected in the contacts panel, the corresponding emails that were sent by this person in the calendar panel were highlighted, revealing the email change frequency between the ego and the selected contact.

Some other studies have investigated on email rhythms. Perer *et al.* [?] analyzed the temporal rhythms, revealing how people interacted with their contacts. Mandic and Kerne [?] visualized the chronological intimacy pattern by using color and shape to characterize the intimacy level of each email. Users are enabled to see how they spend time and energy on people with different intimacy level throughout time. Tyler and Tang [?] focused on the temporal pattern of email usage, enabling users to understand their contacts' response pattern. Xobni [1] and Rapportive [2] provide widgets that show integrated information of users' contacts. In addition to the one-one relationship, several research has devoted to explore the overall interaction patterns lying in collaboration [?] [?] [?], as well as finding a key person among the long contact list [?] [?]. To facilitate managing the contacts, research such as MUSE [?] and ContactMap [?] has provided an automatic grouping of similar contacts.

While the aforementioned visualizations have provided various ways to understand the relationship between people, one cannot know how events have evolved and how different email threads are related to each other. Many work has been proposed to depict the relationship of email threads [?] [?] [?] [?]. ThreadArc [?] adopted a tree visualization technique to display the relationship between emails while maintaining chronology. Related emails were connected by arcs, showing the reply-to relationship as well as how different threads evolved over time. From this visualization, one could observe and compare different threads according to their qualities, such as the size of the thread or the number of responses per message. With visualizations focused on emails, the relationship between the emails can be easily understood. Nevertheless, one has no clue to the relationship between contacts. Rohall *et al.* [?] proposed a reduced-resolution document overview to help users locate the events they are searching for. Frau *et al.* [?] designed a temporal-plot of emails, which enabled users to observe the trends of emails over time and perceive the emails with similar features.

In spite of much having been done in visualizing email archives, with previous visualizations, people could only trace either the relationship of people or the events the email archives presented at a time, but not both of them. Thus, a better visualization that enables people to trace both of the two aspects at the same time is needed.

Some work has proposed similar idea as ours: bring the relationship of people and the evolution of events together. In MUSE [?], different cues were provided to guide users

[1] http://www.xobni.com/

[2] http://rapportive.com/

to explore their own email data. Group cues, name cues, and sentiment cues can be used to remind users about both their relationship with others and some life events. However, while MUSE focused on providing useful cues, they did not visualize how people and events interwove together clearly. Kang *et al.*presented NetLens [?], a system aimed to help people making sense of huge data by providing both identity and topicality. While they had a similar concept as ours, NetLens was more of a query-based system which employed the people panel (represented identity) and the message panel (represented topicality) as separated views, which differed from our integral visualization design that shows both the relationship of people and the evolution of events in one single view.

Except for the work that has been done in the field of email visualization, some work from related fields also has proposed similar concept. Smith and Fiore [?] illustrated the structure of discussion threads in newsgroups, and encoded people with different roles with different glyphs representation. However, one cannot see how the threads of different discussions are related to each other. Moreover, while the members of a topic in a newsgroup will always get each reply thread, email works differently. The recipients could change as email threads go on. For example, one might forward a group message he/she got to other people who did not get the same message. Zhu and Chen [?] presented a communication-garden system, which characterized each thread with its post, participants, and duration as petals, leaves, and flowers. They also use similar technique to decorate each person with his/her messages-posted, threads-participated in, and the time duration that he/she participated in a given topic. While it provided (event) threads and people as complementary information, they did not integrate it in one single view.

2.2 Timeline-based Visualization

As previous work has identified the importance of temporal information in emails [?] [?], we also choose to integral people and event information into a timeline-based visualization. Themeriver [?] used a flow-like visualization to depict the topics changed over time. Textflow [?] extended this idea, and improved on showing the merge and split of topics. Dork *et al.* [?] developed a system of following and exploring large-scale online conversations. Rose *et al.* [**?**] presented a system which linked essential content from streaming data to show how the document changed as the story developed. However, these systems have focused on revealing the topic evolution from mass text data, and did not take the relationship between people into consideration.

2.3 Flow Map Layout

In the field of flow map layout, much work has been proposed to visualize network flow and topology with flow-like smooth edges and minimized visual cluttering. For example, Phan *et al.* [**?**] presented a method for generating flow maps which used the idea of hierarchical clustering to minimize edge crossings. It also prevented distorting node positions while maintaining their relative position to one another. Buchin *et al.* [**?**] introduced a flow map layout algorithm based on spiral trees. By using spiral trees, the algorithm clustered the leaves and smoothly bundled lines. It could also avoid obstacles such as map features or region outlines. While these systems generate satisfying visualization results, we can not simply follow their approaches because flow map layout deals with a different problem when compared with ours. Flow map layout tries to generate smooth edges from a given set of nodes, positions, and edges. The positions of the nodes in flow map layout are given and fixed, and the algorithm only focuses on drawing the edges. However, email data does not have predefined positions as geographic data. Therefore, we need an algorithm that not only generates smooth edges with minimized visual cluttering, but also adjusts and decides the best position for each email node.

Chapter 3

EmailMap Design

In EmailMap, we were mainly interested in integrating the relationship of people (i.e., contacts) and the evolution of events lying in personal email archives. By providing these two aspects in one single view, we hoped to reveal interesting patterns of how people and events are interwoven over time, such as:

- How different people have played parts in different events? Did they devote much time and energy in one single event, or did they participate in several events?
- How contacts interacted in events? Did they join the same event(s) as co-workers, or they have never encountered each other in emails?
- What are the rhythms of interaction within different dyadic relationship between the contact and the ego (i.e., the owner of the email archives)?
- What is the overview of the email archives? How different events evolved and changed over time? What is the time span and when was the busy time of email exchange?

3.1 Dual Design Focus

To the best of our knowledge, most of the email visualization designs focused on only one data dimension, either the relationship of people or the relationship of events. In our first design iteration, we explored the pros and cons of only using the contacts or the email threads as our design focus. It turned out that either of them could only achieve half of our goal. By using the contacts as a design focus it was easier to trace people and the relationship over time but difficult to tell the evolution of events. On the contrary, by using the email threads as the design focus had the opposite effect.

Therefore, we decided to adopt a dual design focus in EmailMap to facilitate tracing either people or events, granting the owner of the email archives (i.e., the ego) the freedom to choose his/her main focus, and providing the two aspects as complementary context information.

In EmailMap, emails are grouped as an event flow shown in the back, and the contacts are depicted as curved tracks going through the emails they have participated in. The horizontal axis represents time progressing from left to right.

3.2 Events as Flow

A straightforward approach to visualize related emails is to adopt the concept of email thread, which is defined as a series of messages sharing the same subject, where the prefixes such as “Re:” and “Fw:” are ignored [?]. However, there are often hundreds of threads in email archives. If we directly visualize these threads, it would be difficult for users to make sense and get a high level concept of the data. Therefore, we view each email thread as a basic event component, and group the email threads into an event flow according to content similarity, participated-contact similarity, and time stamp closeness. In the event flow, each sub-flow represents a sub-event, which can be either a group of threads with a new topic or some follow-up threads with a topic derived from the previous one.

As shown in Fig. 3.1, an email is represented as a circle in its belonged flow with the horizontal coordinate encodes its time stamp. A thread is represented as a gray line going from the first to the last email in the thread. Email threads are grouped together, constructing the event flow. This left-to-right flow goes from the first email to the last, revealing the evolution and relation of events in the past. Thread lines characterize the event flow by showing how many different conversations are going on, and whether there are more long conversations or the opposite. The total length of a thread line indicates the

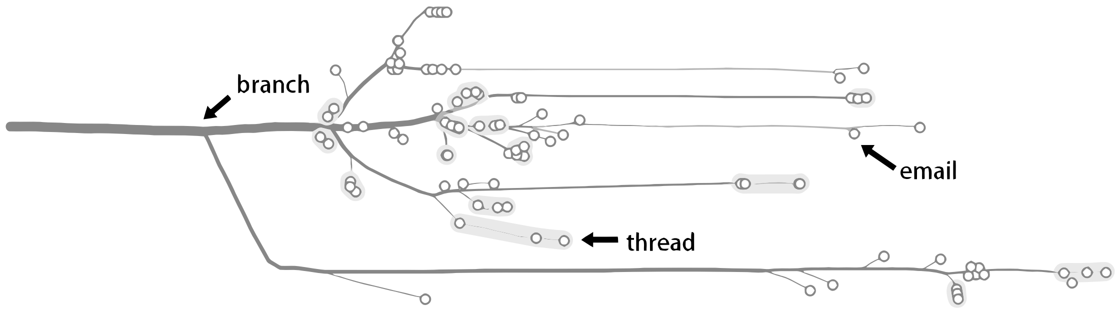


Figure 3.1: An example of an event flow, which consists of three visual elements: branch, email, and thread.

duration of the conversation. Email circles also characterize the thread lines: the intervals between emails show how a given conversation goes on with time – if it is an intensive one or a loose one.

The flow can be split into a number of branches. Each branch shows how one event flow evolve into two or more sub-flows, and when this splitting happened. The thickness of a sub-flow encodes the importance of the flow. Because it is almost impossible to accurately measure the importance of each email message to different users, we define the importance of a sub-flow as the number of the email messages it contains. As email archives could easily go up to hundreds or even thousands of messages, adopting a linear mapping of the weights could cause huge differences of the line thickness. The dramatical change in line thickness could either lead to the unnecessary clutter or an unpleasant non-smooth flow. To address this problem, we define the thickness of a flow to be the square root of the message count.

In order to make the flow more comprehensive, keyword hints are provided and serve as road signs to guide users during the trace of events. An effective way to visualize keywords is to place word clouds next to the branches. However, this method is not suitable in our scenario because branches are often the relatively more complicated area with many sub-flows joining together. Displaying word clouds would thus cause visual clutter. To prevent from cluttering, only the most important keyword extracted from the branching-out sub-flow is placed near the branch. Also, a keyword is placed only when a branch is important, i.e., the weight difference between the sub-flow and its previous flow

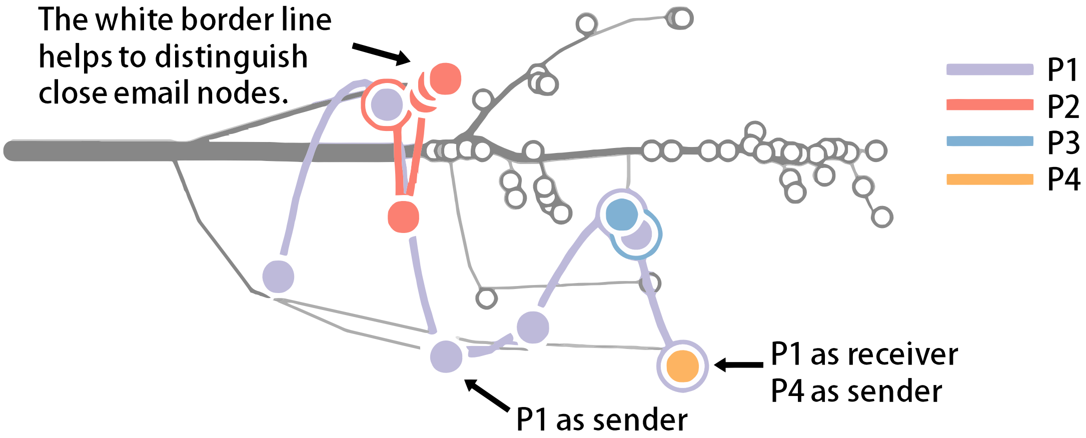

The white border line helps to distinguish close email nodes.
P1
P2
P3
P4
P1 as receiver
P4 as sender
P1 as sender

Figure 3.2: The contacts are represented as curved tracks. When two tracks intersect, the sender is encoded as a solid circle of his/her corresponding color key with white border line, and the receivers are encoded by concentric rings of their corresponding color keys.

exceeds a predefined threshold value, which is allowed for the users to adjust. The size of a keyword is proportional to the branch's weight, which enables users to distinguish different importances of the keywords.

3.3 Contacts as Tracks

To display contact information, a contact is showed as a curved track going through the emails that he/she has participated in. The color is utilized to distinguish different contact tracks.

The email nodes on a contact track are designed as follows: if the person was the sender of the email, the email node is represented as a solid circle of his/her corresponding color key with white border line. The solid-circle design encodes the idea that senders have created the content of the email and thus are the content contributors. The colors filled in an email node maps the concept of the content contributed in the email. The white border line helps to distinguish close email nodes. On the contrary, if the person was the receiver, the email node is depicted as a concentric rings of his/her corresponding color key.

When two or more contact tracks intersect at an email node, the solid circle on the

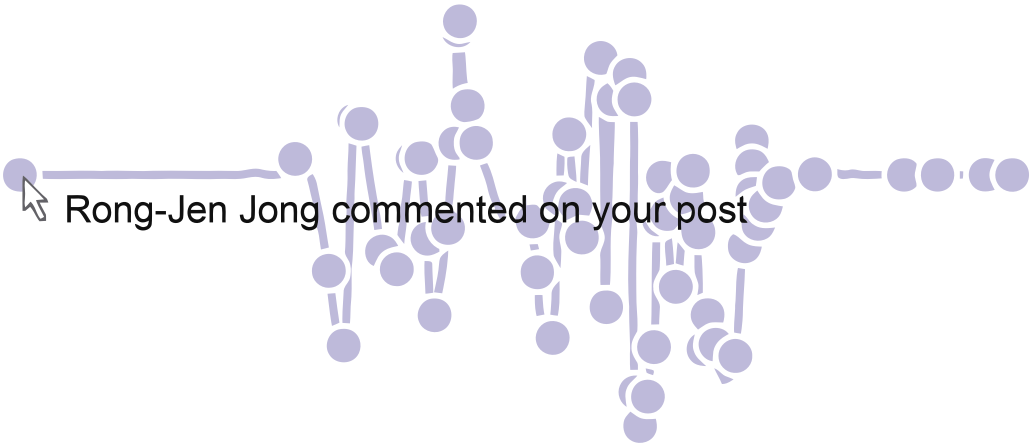


Figure 3.3: Facebook's notification as an example of an active contact.

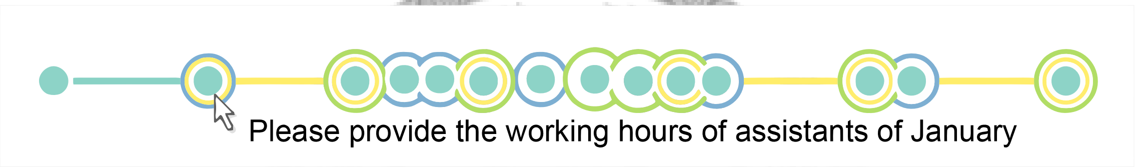


Figure 3.4: A manager as an example of an active contact.

sender's track and the concentric rings on the receivers' track joined together at the node, depicting how these people interact on the email message. Fig. 3.2 shows the interaction of four contacts.

By tracing a single contact track, we can tell the email exchange rhythms between the contacts and the ego. According to the send/receive proportion, we identified three different types of contact.

1. **Active contact**

   Active contacts are those who send emails more than receive emails. An active contact could be a spammer, an automatic system notification center (such as Facebook's notification), a manager, a leader of a project, and so on. Fig. 3.3 shows an example of Facebook's notification being an example of an active contact, who constantly sends email messages about recent activities at Facebook. Fig. 3.4 shows another example of a manager as an active contact, who managed the human resource and asked employees to report related information.

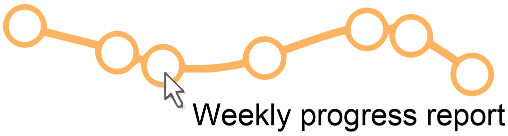


Figure 3.5: A supervisor as an example of a passive contact.

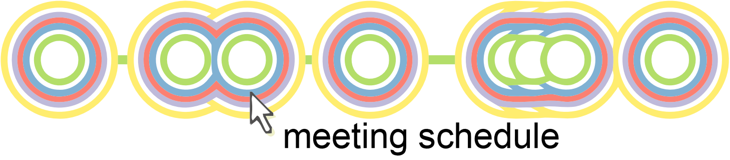


Figure 3.6: A group of students as examples of passive contacts.

2. **Passive contact**

   Passive contacts are those who receive emails more than send emails. A passive contact could be a non-responder, a follower in a project, a supervisor who ask the subordinates to hand in reports often but rarely replies the email, and so on. Fig. 3.5 shows an example of a supervisor as a passive contact, who received his/her subordinates' weekly progress reports from time to time but has never replied. Fig. 3.6 shows another example of five students being passive contacts, who received emails about meeting schedule and followed the schedule announced.

3. **Balanced contact**

   Balanced contacts are those who receive and send emails on a balanced proportion. These contacts participate in the conversation and events in a more bi-directional

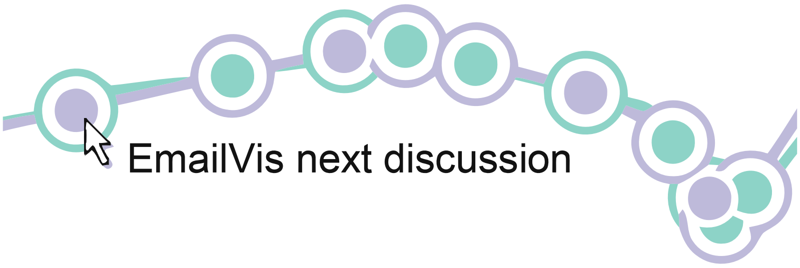


Figure 3.7: Two people working closely on EmailVis project as examples of balanced contacts.

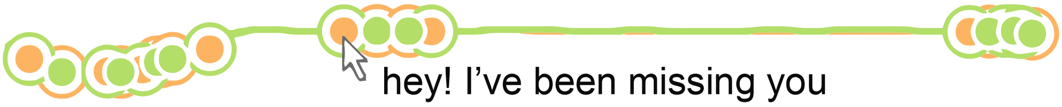


Figure 3.8: Two friends exchange emails from time to time as examples of balanced contacts.

way. They contribute as much content as they receive in the email archives. A balanced contact could be a team member who worked closely with others in a project (Fig. 3.7), a friend whom the ego have contacted with from time to time (Fig. 3.8), a customer who made an inquiry to hotels or travel agencies, and so on.

In addition to the three different types of contacts based on the send/receive ratio, the total length of a contact track indicates the time period of when the contact was connected to the ego through email exchange. Also, by identifying how many different event flow branches a contact track covers, users can tell if the ego has related to the contact on only one event, or across different events in life.

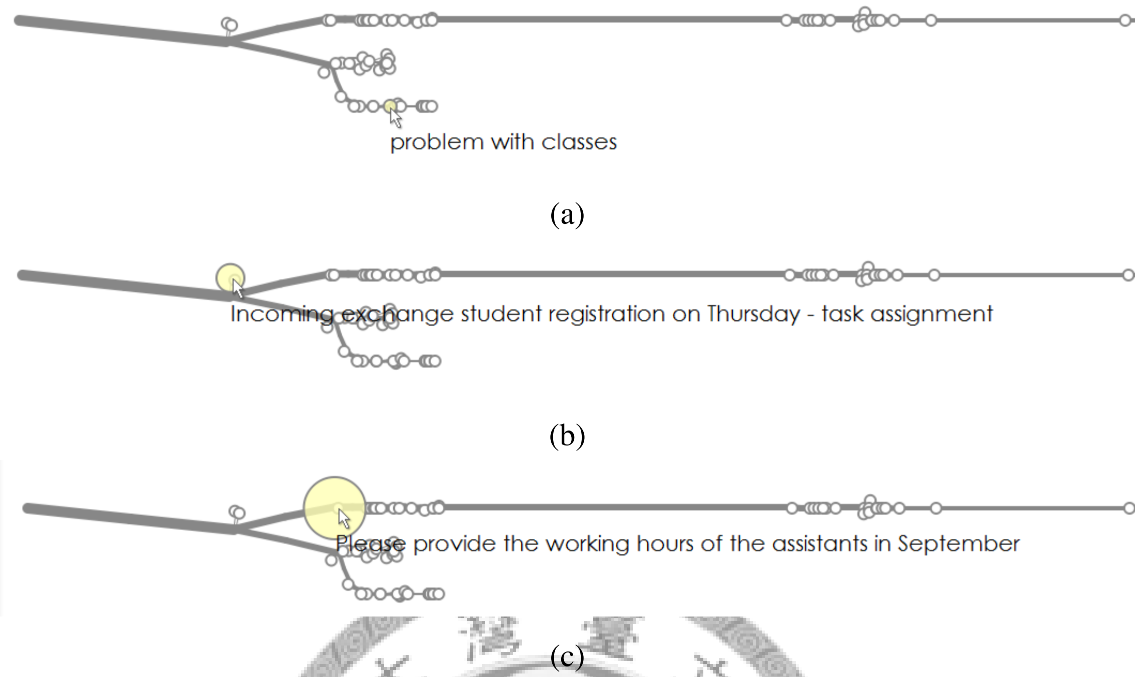
3.4 Interaction

3.4.1 Conversation Thread Interaction

As email archives often contain large amounts of data, the structure of event flow could be composed of many sub-flows and other details. Therefore, the conversation threads were design to be showed according to users' choice. Users can check or uncheck from the control panel to decide if they want to have the conversation information encoded at the moment or not.

3.4.2 Contact Interaction

In email archives, it is common to see hundreds of contacts. However, not all the contacts stand the same importance. To minimize visual clustering and to display the most desired


problem with classes
Incoming exchange student registration on Thursday - task assignment
Please provide the working hours of the assistants in September

(a)

(b)

(c)

Figure 3.9: Three different sizes of email nodes depicting three different numbers of email message receivers.

information, contact tracks are designed as an interactive list. Users can choose from the contact panel and decide which and how many contact tracks to be displayed at a time.

3.4.3 Smart Zooming

Email archives often span several years. To facilitate a holistic overview as well as a more detail local view, zooming and panning are provided. When zooming, only the x-axis which maps the interval of real time data are adjusted. By keeping the y-axis fixed, it prevents the event flow pattern becomes too crowded to see.

3.4.4 Hovering and Double-Clicking

When hovering on an email node, the node is highlighted in yellow, and the subject of the email is displayed. The size of the email node is also adjusted to reflect the number of receivers. A group message is represented as a bigger circle, while an one-to-one message remains the same size. Fig. 3.9 shows examples of hovering on email nodes with different numbers of message receivers. By encoding the number of receivers, the users can distinguish group messages from non-group messages. This dynamic displaying approach also prevent the visualization from getting too complicated.

Double-clicking on an email node shows the content of the email. This enables users to make connections between the event flow structure with their email message memories and experiences. Event flow serves as a structure hint that helps users to organize massive email archives, and each single email message serves as a detail description that enriches the event flow.

Chapter 4

Email Processing

In this section, we describe the algorithms used to process the email archives for visualization.

4.1 Constructing Hierarchical Email Structure

4.1.1 Similarity Method

To construct the event flow, similar emails are grouped together to form a flow for users to trace. As was stated before, email thread is a commonly adopted concept that groups email messages into a series of related messages. Therefore, we first group email messages into email threads, and then apply a similarity analysis technique to group these threads into a hierarchical email structure.

Although general document content similarity analysis has been well studied in the field of information retrieval, it is not suitable for email messages, which include the contact information (such as sender, receivers, and CC receivers) and time stamp information. Simply applying traditional document content similarity algorithms would ignore the similarity lying in contact and time stamp information. Therefore, we developed a similarity measurement approach that integrates the similarity of the content of emails, the participated-contacts of emails, and the time stamp information of emails.

Content similarity.

To calculate content similarity $S_{content}$, we use the widely adopted Salton's TFIDF algorithm [**?**], which determines the weights of feature words of a document by its relative frequency in the corpus.

Formally, the weight can be computed as

$$tfidf_{t,d} = tf_{t,d} \times idf_t, \text{where}$$

$$tf_{t,d} = \frac{n_{t,d}}{\sum_k n_{k,d}}, idf_t = \log \frac{|D|}{|\{d : t \in D_d\}|}, \tag{4.1}$$

where $tf_{t,d}$ indicates the term frequency of a word t in document d, idf_t measures the number of documents in the corpus that contain a word t, $n_{t,d}$ is the number of occurrences of a word t in document d, and $|D|$ is the number of documents in the collection. Then the content similarity is computed by the cosine similarity between two email messages' feature vectors, which is described as

$$S_{content}(d_1, d_2) = \frac{V(d_1) \cdot V(d_2)}{|V(d_1)||V(d_2)|}, \tag{4.2}$$

where $V(d_i)$ denotes the term vector of document d_i.

Participated-contact similarity.

Participated-contact similarity measures the level of overlapping contacts between two email messages. The number of contacts shown in the two messages are used as denominator, whereas the number of appearance of the contacts shown in both the two messages are used as the molecular. For example, if contacts A, C, and E are shown in message 1, and contacts A, and B are shown in message 2, the participated-contact similarity will be 2/5. The formula is described as:

$$S_{contact}(d_1, d_2) = \frac{|2P(d_1 \cap d_2)|}{|P(d_1)| + |P(d_2)|}, \tag{4.3}$$

where $P(d_i)$ is the number of participated contacts of message d_i.

Time stamp difference.

Time stamp difference measures the time interval between two email messages. As email is by nature a data type that encodes time information, people mentally classify their email messages not only by the content and the contact, but also the time period. The longer the interval between two email messages is, the less similar they are in terms of time. Time stamp difference is modelled as a time penalty over the content similarity and participated-contact similarity, and is defined as:

$$P_{time}(d_1, d_2) = \begin{cases} 1 - \frac{T_{diff}(d_1,d_2)}{T_{threshold}}, & \text{if } T_{diff}(d_1, d_2) < T_{threshod} \\ 0, & \text{otherwise.} \end{cases}$$

where $T_{diff}(d_i, d_j)$ is the time difference between message d_i and message d_j, and $T_{threshold}$ is a predefined threshold value that relates to the overall time span of the email archives. This threshold can also be set to a huge number if the user prefers to ignore the time factor. In our implementation, we use different threshold values based on different data sizes and characteristics.

Email message similarity is calculated by the weighted combination of the two similarity measures and the time penalty:

$$S_{email}(d_1, d_2) = P_{time}(d_1, d_2)((1 - w)S_{content}(d_1, d_2) + wS_{contact}(d_1, d_2)). \tag{4.4}$$

In our implementation, we use $w = 0.5$.

4.1.2 Hierarchical Email Clustering

Then, we describe how we group email threads into an event flow. The event flow is designed to enable users to browse similar email threads as the components of a high-level event concept. Adopting flat clustering would eliminate the different levels of similarity, which could provide useful information and contribute to a richer resulting event flow. Therefore, we adopted hierarchical clustering as our basic concept, which enables the dynamic control of final grouping numbers. This flexibility could be used to adapt email archives with different attributes (with many long or short conversations), providing a cluttering-minimized and meaning-preserved visualization.

4.1.3 Hierarchical Email Structure

Binary tree is a widely-adopted structure to present the structure of hierarchical clustering. However, if we simply apply a binary tree as our structure, we would fail to encode the

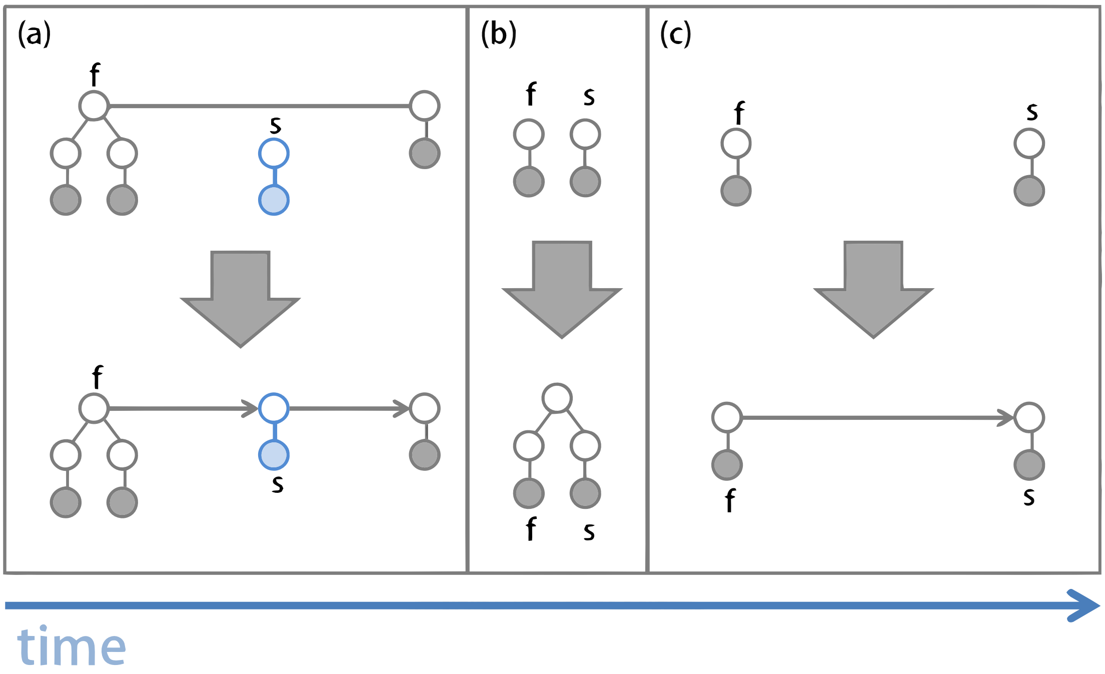


Figure 4.1: The three clustering conditions, (a) temporally overlap, (b) non-overlap, temporally close to each other, and (c) non-overlap, temporally far away from each other.

time data of emails, which is a prominent feature that should be preserved and well-considered. Considering the time factor, we categorize grouping two similar email threads into three conditions: (1) temporally overlap, (2) non-overlap, temporally close to each other, and (3) non-overlap, temporally far away from each other. Fig. 4.1 depicts the three conditions.

1. *Temporally overlap.* Two similar email threads with temporally overlap (i.e., the latter one starts while the former one has not ended yet) might indicate that the latter one was triggered by the former one. To encode the possible derived-from relation, we add the latter one as a branch from the former one. In other words, the latter one is visually encoded as an event derived from the former one.

2. *Non-overlap, temporally close.* When two email threads have no overlap and are temporally close, it is relatively vague if one is derived from the other. Therefore, we create a new branch node linked to both of them to depict that the event has split into two subevents closely related to each other.

3. *Non-overlap, temporally apart.* When two email threads have no overlap and are

temporally far away from each other, it might be the case that these two threads are related to another bigger event. Therefore, we connect both threads to another trunk.

The second and third cases are distinguished by a time threshold parameter. We set this parameter to 1-day as default.

4.1.4 Connecting to the Flow Root

As email archives could be messy and diversified, it is likely that not all the email threads will be grouped and connected together through the hierarchical clustering algorithm. Often, we would end up with several hierarchical groups. To connect these group roots to the final flow root and prevent from visual cluttering, we connect the important groups' roots directly to the flow root while those non-important groups' roots are first connected to each other in time order, and then the earliest group root is connected to the flow root. In our implementation, the important groups is identified by a weight threshold value that relates to the email archives' size.

4.2 Keyword Extraction

In order to generate keyword hints, a set of keywords are extracted from the email archives. As email subjects usually summarize and give good hints about what the message is about, keywords are extracted from the message subjects. As the event flow grows, keyword lists are updated to its ancestor nodes. It ensures that the keyword list contained in each node describes its descendant properly.

4.3 Contact Processing

The major issues when processing the contacts are that people might have multiple email addresses. As people could have different displaying names of these email addresses (e.g., "Jeremy Lin" and "Shu Hao Lin"), and different people could also have exactly the same displaying name, to precisely identify email addresses that belong to the same person is almost impossible.

In EmailMap, we choose to regard each distinct email address as an independent contact flow. The advantages of this design are as follows. First, users will not get confused or misguided by wrongly aggregated email addresses. Second, users can see more insights about the contacts, such as how a contact shift from one mail address to another, how a contact use different email addresses in dealing with different events, and so on. Third, users can still track all the email addresses belonging to the same contact by selecting on the contact control panel.

國立臺灣大學
敦品勵學・愛國愛人

Chapter 5

Computing Layout

5.1 Computing Event Flow Layout

5.1.1 Input Hierarchical Email Structure

To draw the event flow with smooth curves for aesthetics and readability, we applied an optimization technique to the hierarchical email structure obtained from the previous section. To compute the layout, we formulate a number of visual constraints into an objective function, and find the unknown positions of nodes that minimize the function.

Formally, we denote the input hierarchical email structure by $\mathbf{T} = \{\mathbf{V}, \mathbf{E}\}$, where $\mathbf{V} = \{\mathbf{v}_1, ..., \mathbf{v}_n\}$ is a set of n nodes, $\mathbf{v}_i = (\mathbf{v}_{i,x}, \mathbf{v}_{i,y}) \in \mathbb{R}^2$, and $\mathbf{E}$ is the connecting edges. The input structure has the following four types of nodes:

- **Root node.** The root of the event flow with no parent node.
- **Email node.** The node which stands for an email message, with time stamps to indicate when the email was sent.
- **Extend node.** The node which was used when constructing the hierarchical email structure.
- **Structure node.** The node which is generated in the initial layout process and is used to maintain the important feature of the initial layout.

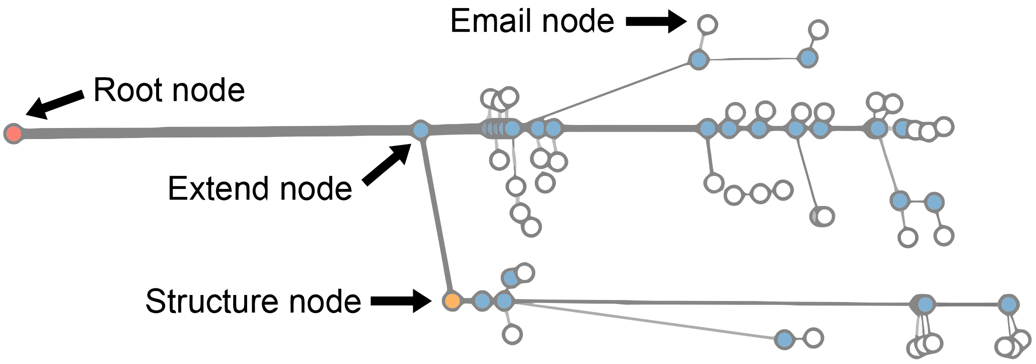


Figure 5.1: Displaying an example of the input structure for the optimization: the root node is colored in red, the email nodes are colored in white, the extend nodes are colored in blue, and the structure node is colored in orange. In this figure, the positions of nodes are adjusted in order to clearly display all four types of nodes.

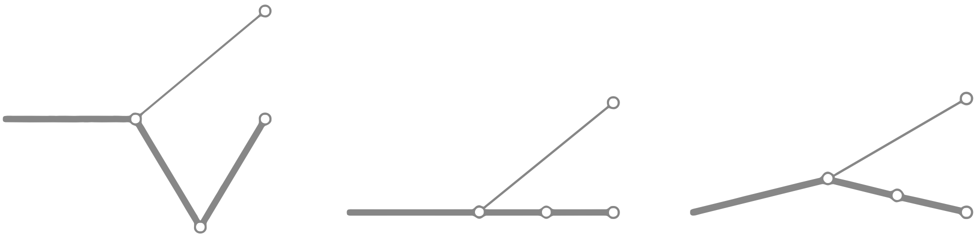
Fig. 5.1 is an example of input structure for the optimization process with the above four types of nodes.

To obtain a smooth flow layout, we iteratively subdivide an edge and add a subdivision node at the center of the edge if it is longer than a predefined length l_ϵ during the optimization process:

- **Subdivision node.** The node which is added during the optimization process.

Therefore, the resulting event flow structure has five types of nodes in total, with four types originated from the input structure, and one type generated during the subdivision process. We denote the sets of the above types of nodes by $\mathbf{V}_r, \mathbf{V}_e, \mathbf{V}_{ext}, \mathbf{V}_{str}, \mathbf{V}_{sub}$, respectively.

All the nodes are considered during the optimization process, but only email nodes are displayed in the final visualization result. In addition, to encode the thickness of the flow, each node has a weight ($w_{\mathbf{v}_i}$) indicating the number of email nodes of the subtree rooted at the node.



(a) (b) (c)

Figure 5.2: The results of optimizing the energy function Ω with and without the weight $w_{\mathbf{v}'_\mathbf{c}}{}^2$. Notice that the thickness of each branch indicates the value of $w_{\mathbf{v}'_\mathbf{c}}$. (a) The input tree structure. (b) The result with $w_{\mathbf{v}'_\mathbf{c}}{}^2$. (c) The result without $w_{\mathbf{v}'_\mathbf{c}}{}^2$.

5.1.2 Objective Function

We introduce a number of constraints that can capture the properties of a good flow, and compute a set of node positions $\mathbf{V}'$. Specifically, we model the constraints as smoothness cost, structure cost, occlusion cost, and time stamp cost.

Smoothness cost. To make the event flow as smooth as possible, we encourage the connecting edges to have similar directions. Therefore, we define a smoothness cost for each node which has a parent node and at least one child node. Moreover, if an edge branches into a thicker edge and a thinner edge, it is more pleasing if the thicker edge is straighter, as it would allow users to easily trace the main branch flows. To implement the idea, we minimize:

$$\Omega_s = \sum_{\mathbf{v}'_i \in \mathbf{V}'_\varepsilon} \sum_{\mathbf{v}'_c \in C(\mathbf{v}'_i)} w_{\mathbf{v}'_\mathbf{c}}{}^2 |s_{ip}(\mathbf{v}'_i - \mathbf{v}'_{P(\mathbf{v}'_i)}) - s_{ic}(\mathbf{v}'_c - \mathbf{v}'_i)|^2, \tag{5.1}$$

where $\mathbf{V}'_\varepsilon$ is the set of nodes which have a parent node and at least one child node, $C(\mathbf{v}'_i)$ is the set of $\mathbf{v}'_i$'s children, $P(\mathbf{v}'_i)$ is the parent node of $\mathbf{v}'_i$, $s_{ip} = |\mathbf{v}_c - \mathbf{v}_i|/(|\mathbf{v}_c - \mathbf{v}_i| + |\mathbf{v}_i - \mathbf{v}_{P(\mathbf{v}_i)}|)$ and $s_{ic} = 1 - s_{ip}$ eliminate that influences of edge lengths and ensure that only the directions are taken into consideration. The weight $w_{\mathbf{v}'_\mathbf{c}}{}^2$ encourages the thicker branch to be straightened more. Fig. 5.2 shows the comparison of the optimization results with/without the weight $w_{\mathbf{v}'_\mathbf{c}}{}^2$. Notice that with the weight, the main branch would be straightened more (Fig. 5.2(b)).

Structure cost. The structure nodes should locate around their initial positions to preserve

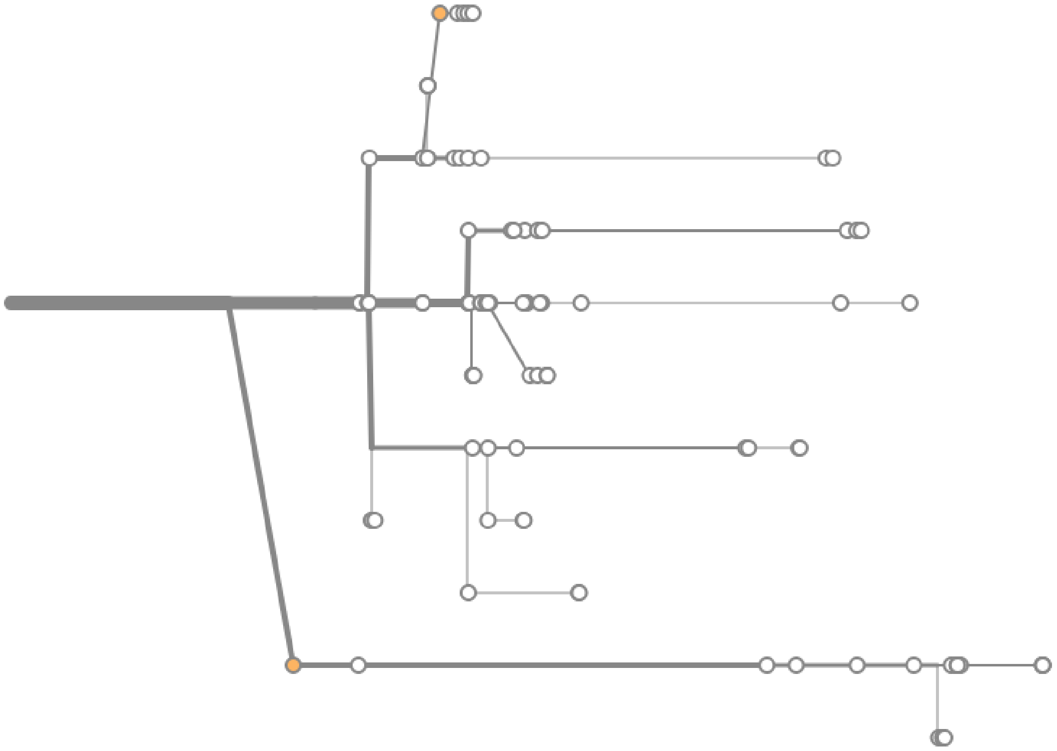


Figure 5.3: The initial layout of a set of real data. Email nodes are colored in white. The structure node colored in orange is displayed in order to show the effect of the structure cost.

important structural features. Therefore, we penalize the distance between the nodes' x-coordinates and their target x-coordinates and the nodes' y-coordinates and their target y-coordinates. Specifically, the structure cost is defined as:

$$\Omega_{str} = \sum_{\mathbf{v}'_i \in \mathbf{V}'_{str}} |\mathbf{v}'_{i,x} - X_{\mathbf{v}_i}|^2 + |\mathbf{v}'_{i,y} - Y_{\mathbf{v}_i}|^2, \tag{5.2}$$

where $X_{\mathbf{v}_i}$ is the target x-coordinates, $Y_{\mathbf{v}_i}$ is the target y-coordinates. Notice that the structure cost is only used to constraint the structure nodes.

As we can see by comparing Fig. 5.3 and Fig. 5.4, while the smoothness term worked for making the flow more smooth, it also caused edge crossing as the important structure feature (the orange node) was not maintained during the optimization process. By adding the structure cost in Fig. 5.5, the crossing was prevented.

Occlusion cost. The email nodes should be prevented from occlusion by other nodes for readability. In addition, it is more visually pleasing if the edges are clearly separated.

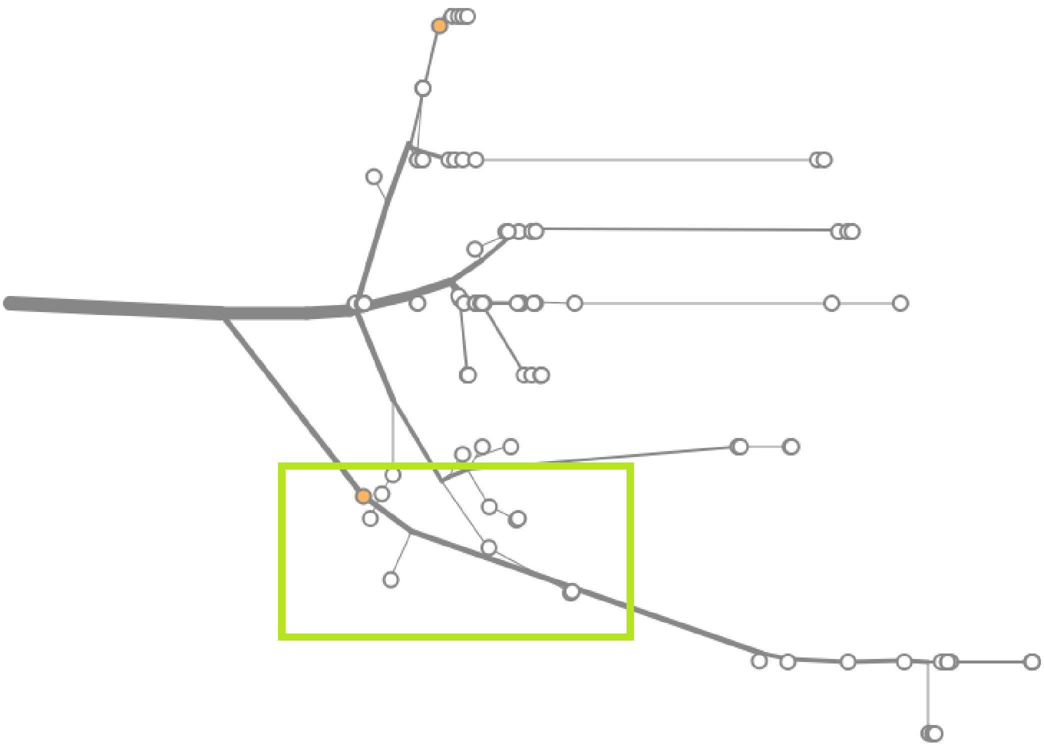


Figure 5.4: The optimization result of Fig. 5.3 by adopting the weights as follows: $w_s = 1, w_{str} = 0, w_o = 0$, and $w_t = 0$.

Therefore, the occlusion cost is designed to ensure that all nodes keep a predefined distance d_ϵ from other nodes. Specifically, the occlusion cost is defined as:

$$\Omega_o = \sum_{\mathbf{v}'_i \in \mathbf{V}'_e} \sum_{\mathbf{v}'_j \in \mathbf{V}'_e, \mathbf{v}'_i \neq \mathbf{v}'_j} \Omega_o(\mathbf{v}'_i, \mathbf{v}'_j), \tag{5.3}$$

where

$$\Omega_o(\mathbf{v}'_i, \mathbf{v}'_j) = \begin{cases} (d_\epsilon - |\mathbf{v}'_i - \mathbf{v}'_j|^2), & \text{if } |\mathbf{v}'_i - \mathbf{v}'_j|^2 < d_\epsilon \\ 0, & \text{otherwise.} \end{cases}$$

Notice that the occlusion cost is only used to constraint the email nodes.

Time stamp cost. The email nodes should locate on the position where the temporal coordinate corresponds to their sent time stamps. Therefore, we penalize the distance between the nodes' x-coordinates and their target x-coordinates. Specifically, the time stamp cost is defined as:

$$\Omega_t = \sum_{\mathbf{v}'_i \in \mathbf{V}'_e} |\mathbf{v}'_{i,x} - X_{\mathbf{v}_i}|^2, \tag{5.4}$$

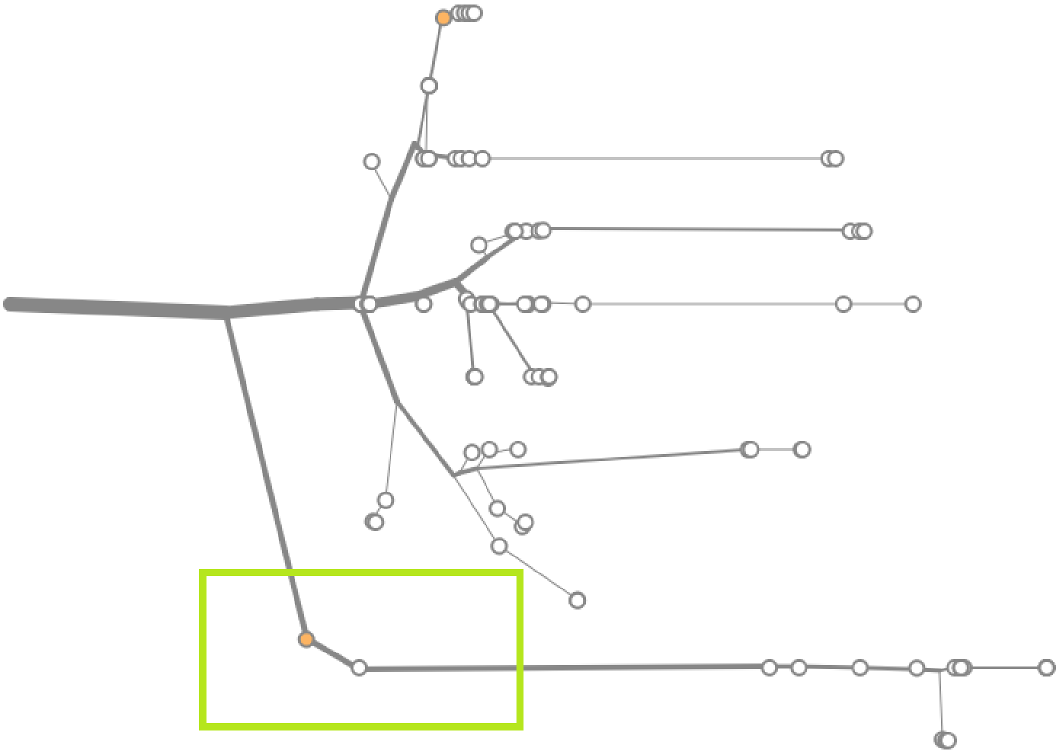


Figure 5.5: The optimization result of Fig. 5.3 by adopting the weights as follows: $w_s = 0.1, w_{str} = 10, w_o = 0$, and $w_t = 0$.

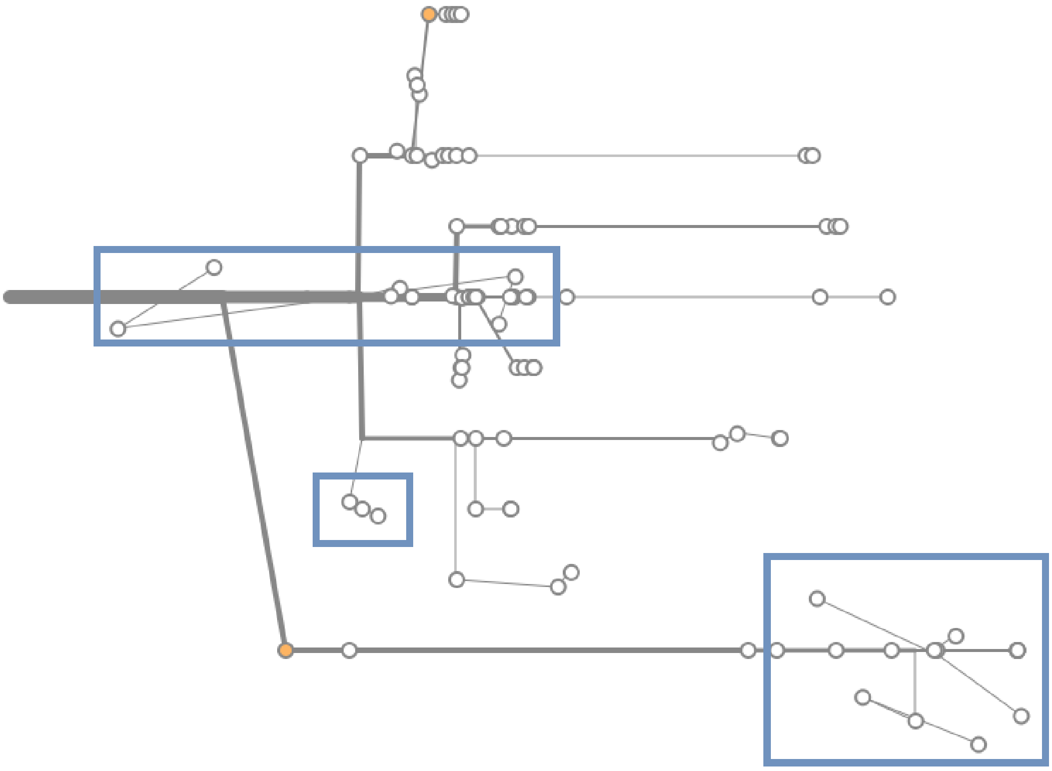


Figure 5.6: The optimization result of Fig. 5.3 by adopting the weights as follows: $w_s = 0, w_{str} = 0, w_o = 0.1$, and $w_t = 0$.

where $X_{\mathbf{v}_i}$ is the target x-coordinates. Notice that the time stamp cost is only used to constraint the email nodes.

As we can see by comparing Fig. 5.3 and Fig. 5.6, while the occlusion term prevented the occlusion of email nodes, it also moved some email nodes' x-coordinates too much to preserve their time stamp information. By adding the time stamp cost in Fig. 5.7, the time stamp information was kept.

The total objective function of the optimization is a weighted sum of the cost terms defined above:

$$\Omega = w_s\Omega_s + w_{str}\Omega_{str} + w_o\Omega_o + w_t\Omega_t. \tag{5.5}$$

The weights are determined by experimenting with different values and inspecting the results. Fig. 5.8 is the optimization result of Fig. 5.3 by adopting the weights as follows: $w_s = 1, w_{str} = 1000, w_o = 500$, and $w_t = 1000$.

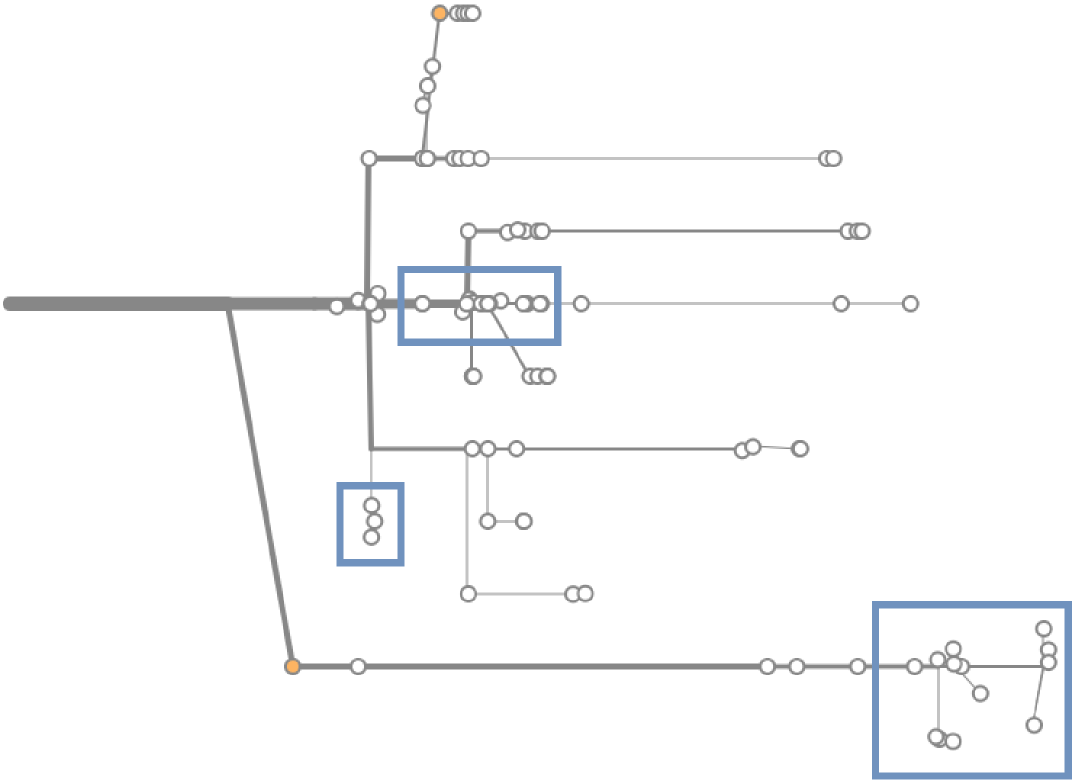


Figure 5.7: The optimization result of Fig. 5.3 by adopting the weights as follows: $w_s = 0, w_{str} = 0, w_o = 10$, and $w_t = 100$.

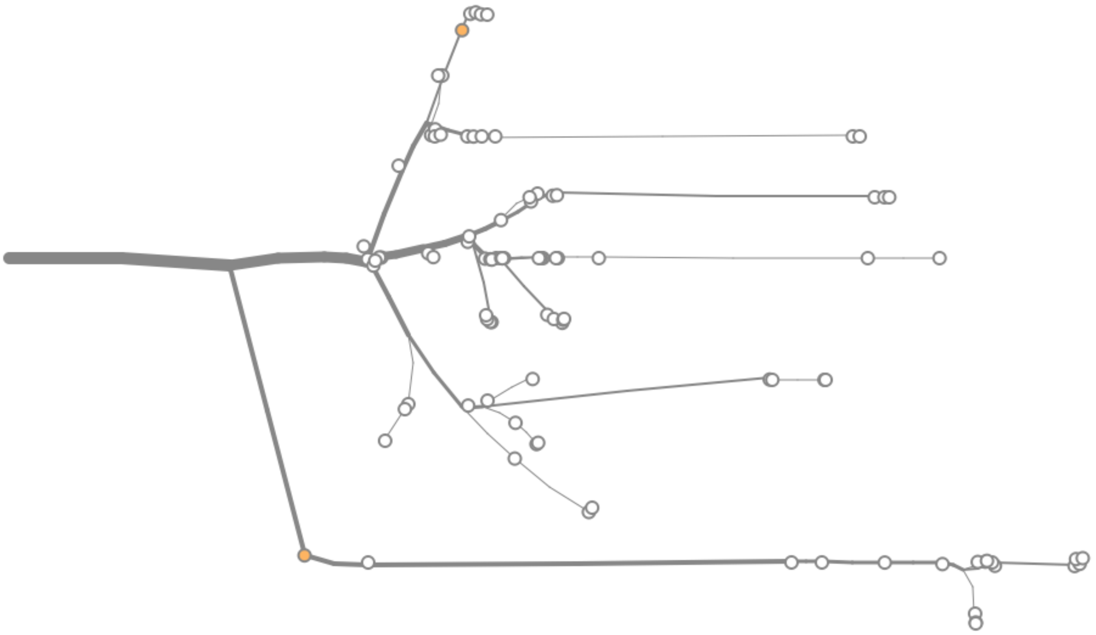


Figure 5.8: The optimization result of Fig. 5.3 by adopting the weights as follows: $w_s = 1, w_{str} = 1000, w_o = 500$, and $w_t = 1000$.

5.1.3 Optimization

In this section, we describe how we minimize the objective function Ω to solve for the positions of all the nodes, say $\mathbf{V}'$. The steepest descent method is applied to minimize the objective function, which iteratively moves the positions with a lower energy. Formally, at each iteration the nodes' positions are updated as $\mathbf{V}'_{t+1} = \mathbf{V}'_t - \epsilon\Delta\Omega(\mathbf{V}'_t)$, where ϵ scales the step of the gradient vector $\Delta\Omega$. To find an adequate ϵ, a step-doubling line search strategy is adopted. Starting from the point in $\mathbb{R}^{2n}$ defined by $\mathbf{V}'_{i-1}$, it takes steps along the gradient direction, and doubling step length until the objective function does not decrease, then choose the one with lowest energy.

Initial layout. The iterative optimization requires an initial guess. In our proof-of-concept implementation, we generate the initial layout by adopting a rule-based strategy. The rules capture a number of aesthetic criteria, and are summarized as follows. First, except for the structure nodes that are added during the initial layout process, only the y-coordinates are assigned values. The x-coordinates are already decided and mapped to the email time stamp information when constructing the hierarchical email structure. Second, the layout algorithm follows a depth-first search (DFS) order to assign each node's y-coordinate while the deepest node is defined as the child node with the most weight. If two or more child nodes have exactly the same weights, the deepest child is defined as the child node with the longest subtree. However, when starting from the flow root, the child node which leads a subsequence of non-important subtrees as described in 4.1.4 will be the last child node to be assigned. This is the only exception. Third, when deciding the y-coordinate of a given node, the algorithm searches for a y-coordinate that has not been occupied (in terms of the node's x-coordinate). It starts from its parent node's y-coordinate, and then towards the upper side and the lower side in turn. It aims to find a y-coordinate where no other node is placed and as close to its parent node's y-coordinate as possible. Last, when a given node's both coordinates exceed certain distance from its parent node's coordinates, a structure node is added. The orange nodes in Fig. 5.1 and Fig. 5.3 are both examples of structure nodes.

Dealing with large-scale email data. Users usually have a large amount of email messages over a long time period in their archives. As a consequence, we can expect an event flow with a large amount of nodes to optimize. To solve a large amount of nodes' positions is technically intractable and inefficient. Hence, we introduce two strategies to deal with this problem: coarse-to-fine optimization and sliding window optimization.

The goal of the first strategy is to solve a rough high-level event flow layout by optimization, followed by subdividing the input hierarchical email structure to get a finer structure and computing its layout. Specifically, we first apply the optimization to the input hierarchical email structure, denoted by $\mathbf{T}_0$, to compute $\mathbf{T}_0'$. Then, we subdivide $\mathbf{T}_0'$ by inserting one subdivision node at the center of each edge whose length is longer than the predefined length threshold l_ϵ, and get a finer structure $\mathbf{T}_1$. We then optimize the finer structure to get $\mathbf{T}_1'$ and iterate the process until no subdivision node is inserted.

The input hierarchical email structure may contain hundreds of nodes and thus is inefficient even when the coarse-to-fine strategy is applied. Therefore, rather than globally solving the optimization problem, we propose a sliding window optimization strategy. Specifically, starting from a certain temporal coordinate (e.g., the root node), we compute the energy only for the nodes located in a local temporal window and then shift the window forward and backward. The window size is inversely proportional to the subdivision level of the event flow. The two strategies are applied together to optimize an input hierarchical email structure. In our experiment, although it is not guaranteed to get a globally optimal solution, the coarse-to-fine strategy works well together with the sliding window strategy for most input data.

5.2 Computing Contact Tracks Layout

A contact track is a curved track going through the emails that the contact has participated in. The curved track is generated by following a similar optimization approach as the previous section.

Formally, we denote the input contact track by $\mathbf{T\_ct} = \{\mathbf{V}, \mathbf{E}\}$, where $\mathbf{V} = \{\mathbf{v}_1, ..., \mathbf{v}_n\}$ is a set of n nodes, $\mathbf{v}_i = (\mathbf{v}_{i,x}, \mathbf{v}_{i,y}) \in \mathbb{R}^2$, and $\mathbf{E}$ is the connecting edges.

Each input contact track have the following two types of nodes:

- **Email node.** The node which stands for an email message, with time stamps to indicate when the email was sent.

- **Subdivision node.** The node which is added by iteratively subdivide an edge and add a subdivision node at the center of the edge if it is longer than a predefined length l_ϵ.

We denote the sets of the above types of nodes by $\mathbf{V}_{ct\_e}$ and $\mathbf{V}_{ct\_sub}$ respectively.

Smoothness cost. To make the contact track as smooth as possible, we follow the similar idea of the smoothness term defined for the event flow. However, as all the edges of contact tracks have the same thickness, and the length proportions of neighboring edges are not as important for contact tracks, we simply encourage the connecting edges to have similar directions.

To implement the idea, we minimize:

$$\Omega_{ct\_s} = \sum_{\mathbf{v}'_i \in \mathbf{V}'_\varepsilon} \sum_{\mathbf{v}'_c \in C(\mathbf{v}'_i)} |(\mathbf{v}'_i - \mathbf{v}'_{P(\mathbf{v}'_i)}) - (\mathbf{v}'_c - \mathbf{v}'_i)|^2, \tag{5.6}$$

where $\mathbf{V}'_\varepsilon$ is the set of nodes which have a parent node and at least one child node, $C(\mathbf{v}'_i)$ is the set of $\mathbf{v}'_i$'s children, $P(\mathbf{v}'_i)$ is the parent node of $\mathbf{v}'_i$.

Deviation cost. The contact track goes through the email nodes that have been positioned during the event flow layout process. Therefore, the email nodes should be changed as less as possible during the contact track optimization to maintain the original event flow layout.

Specifically, the deviation cost is defined as:

$$\Omega_{ct\_d} = \sum_{\mathbf{v}'_i \in \mathbf{V}'_{ct\_e}} |\mathbf{v}'_{i,x} - X_{\mathbf{v}_i}|^2 + |\mathbf{v}'_{i,y} - Y_{\mathbf{v}_i}|^2, \tag{5.7}$$

where $X_{\mathbf{v}_i}$ is the target x-coordinates, $Y_{\mathbf{v}_i}$ is the target y-coordinates. Notice that the structure cost is only used to constraint the email nodes.

As we can see by comparing Fig. 5.9 and Fig. 5.10, while the smoothness term worked for making the contact track more smooth, it also moved the email nodes too much and

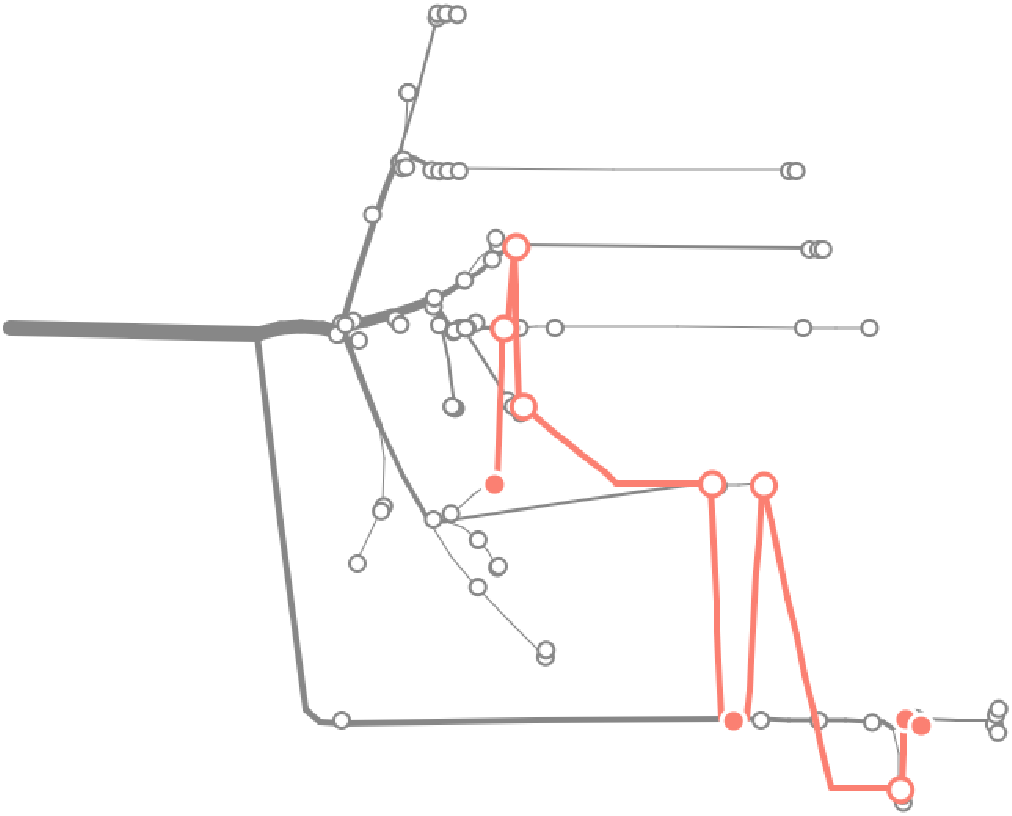


Figure 5.9: An input structure of a contact track.

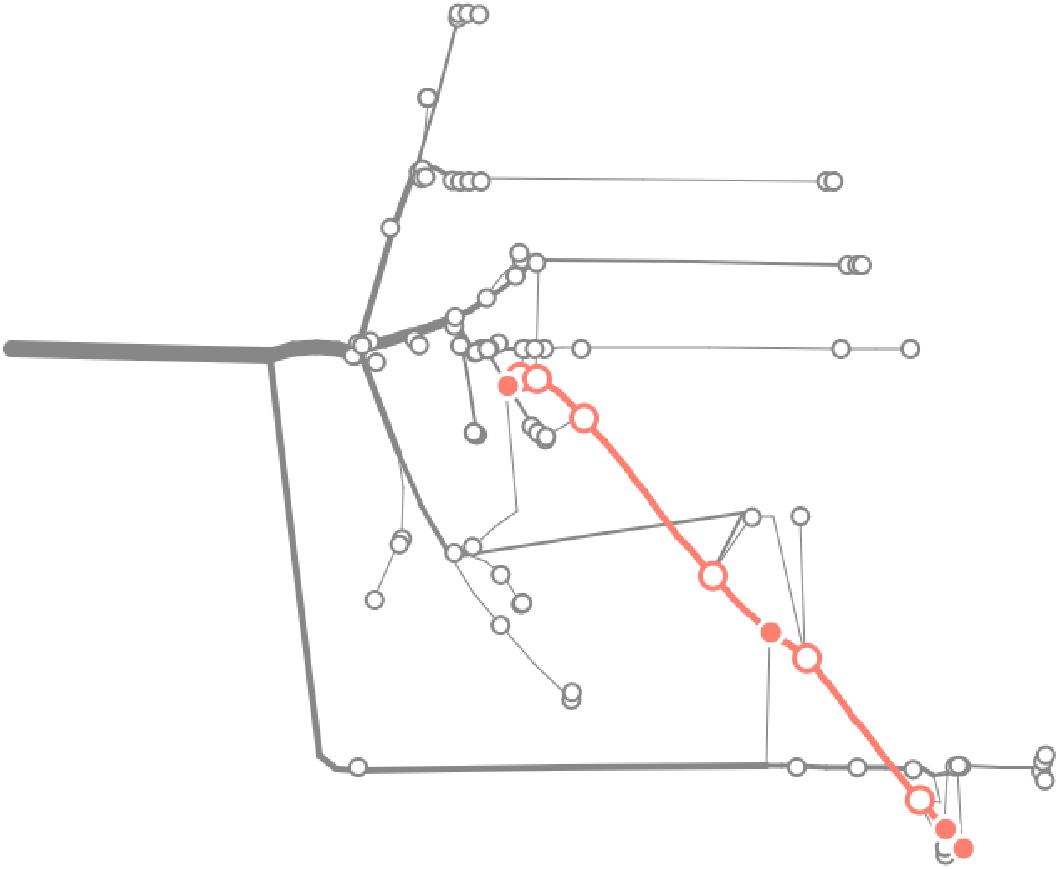


Figure 5.10: The optimization result of Fig. 5.9 by adopting the weights as follows: $w_{ct\_s} = 100$ and $w_{ct\_d} = 0$.

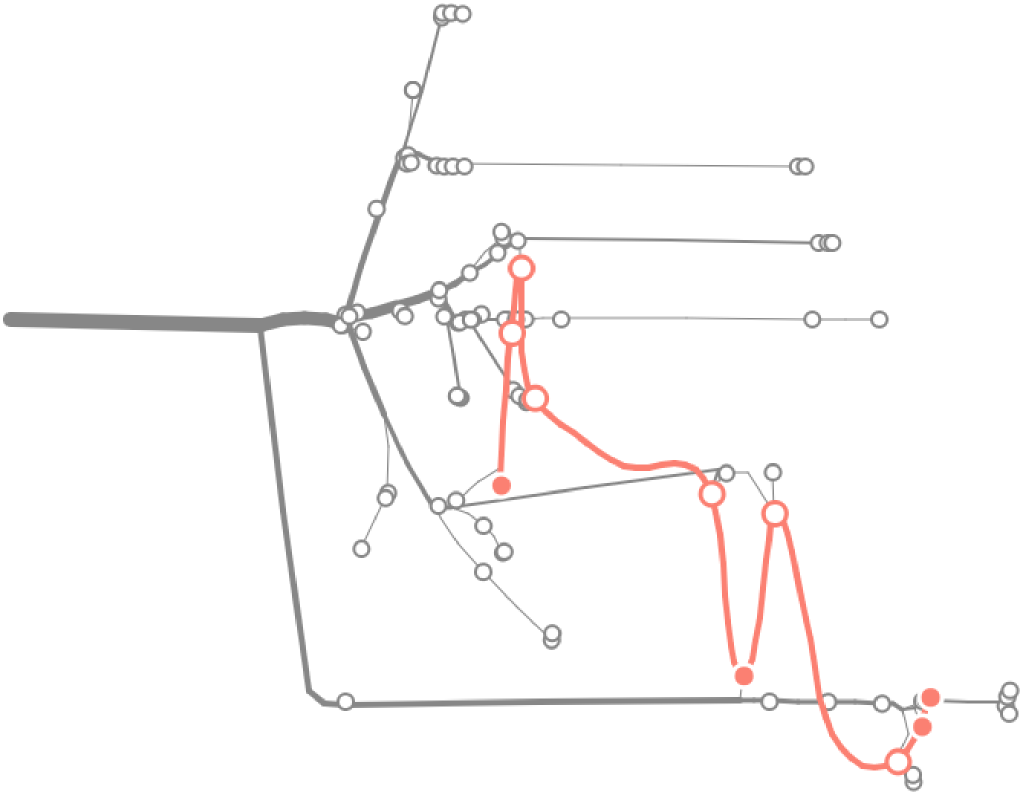


Figure 5.11: The optimization result of Fig. 5.9 by adopting the weights as follows: $w_{ct\_s} = 0.1$ and $w_{ct\_d} = 100$.

distorted the already-optimized event flow structure. By adding the deviation cost in Fig. 5.11, the event flow was better preserved.

The total objective function for contact track optimization is a weighted sum of the cost terms defined above:

$$\Omega_{ct} = w_{ct\_s}\Omega_{ct\_s} + w_{ct\_d}\Omega_{ct\_d}. \tag{5.8}$$

The weights are determined by experimenting with different values and inspecting the results.

國立臺灣大學 · 敦品勵學 · 愛國愛人

Chapter 6

Results and Discussion

6.1 Case Study

6.1.1 Case 1: EmailMap Project

Fig. 6.1 shows the visualization of all the mails related to EmailMap project. As can be seen from it, this project started around June, 2011 and lasted until the end of March, 2012 (the deadline of InfoVis 2012). The frequency of message exchanging has increased since mid-November, 2011. A dramatic rise was found at the end of March, 2012.

This project has one main event flow lying in the center and some sub-event flows branching out on different points of time. The main event flow contains the discussion of the design and the implementation of EmailMap overtime, while the others include the discussion of some noticible references. For example, the hovered on message was when we were trying out Xoboni [1].

Several participants were highlighted. The orange, the blue and the red contact tracks indicate that these contacts were involved for a relative longer time compared to the pink one and the purple one. These three all represent professors who gave us advises from time to time, but did not participated intensely in the work. The pink and the purple contacts were other researchers we consulted and have provided some related information.

Fig. 6.2 shows how the major two researchers have been collaborating closely together on the EmailMap project. The hovered on email node with a relative small size of highlighted circle shows that the discussion was between a small group of people, instead of

[1] http://www.xobni.com/

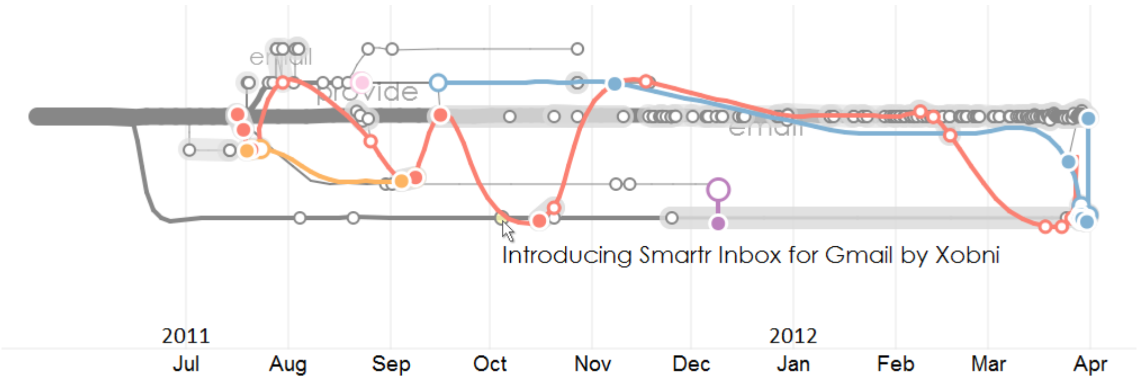


Figure 6.1: The visualization of EmailMap project. Five different contact tracks are displayed. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1, w_{str} = 100, w_o = 500$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: w_{ct_s} = 1 and $w_{ct\_d} = 500$.

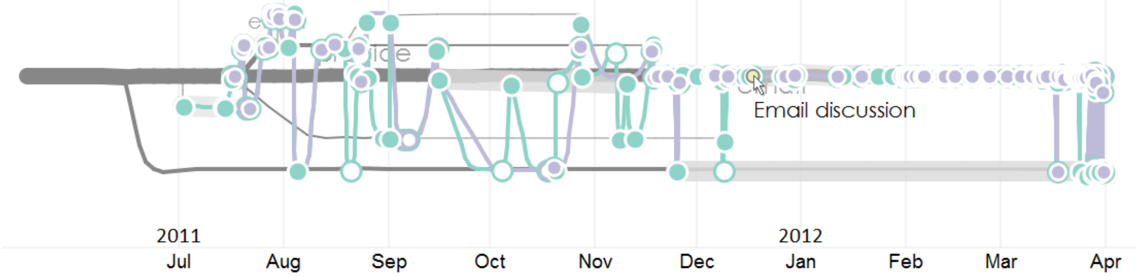
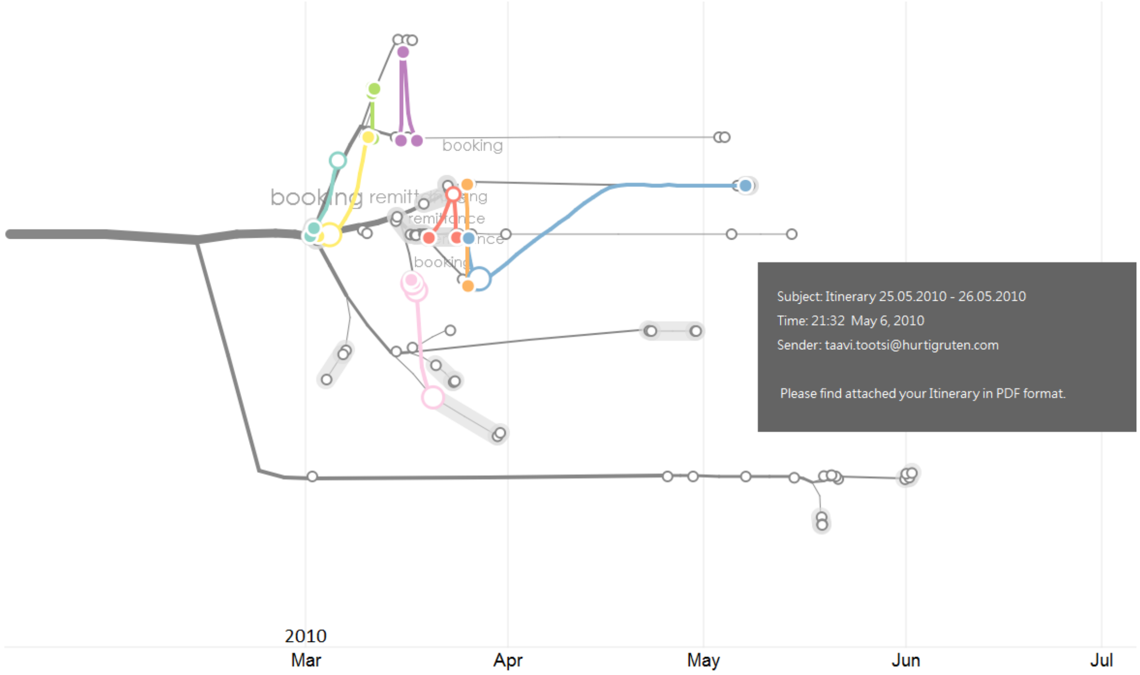


Figure 6.2: The visualization of EmailMap project. Two different contact tracks are displayed. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1, w_{str} = 100, w_o = 500$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: w_{ct_s} = 1 and $w_{ct\_d} = 500$.



Subject: Itinerary 25.05.2010 - 26.05.2010
Time: 21:32 May 6, 2010
Sender: taavi.tootsi@hurtigruten.com

Please find attached your Itinerary in PDF format.

Figure 6.3: The visualization of planning a trip to Norway. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1, w_{str} = 1000, w_o = 500$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: w_{ct_s} = 1 and w_{ct_d} = 100.

two major researchers leading several other team members.

6.1.2 Case 2: A Trip to Norway

Fig. 6.3 depicts an Norway travel plan. The most busy time of message exchanging was in March, 2010, when the traveller started to organize the itinerary and book tickets and hostels. Keyword hints such as “booking” and “remittance” can be seen.

Seven short contact tracks represent the booking confirmation emails as well as the inquiring emails about transportation, available rooms, and prices. These almost vertical tracks portray the quick responses from most of the hostels and transportation companies. The longer blue contact track shows one booking process took longer than the others.

It also reveals the traveller’s pattern of planing a trip: he/she booked and confirmed most of the things within one month (March, 2010). Then, a smaller amount of email traffic came right before going on the trip to Norway (at the end of June, 2010). We can also see that there is one message content displayed after double clicking on an email node.

6.1.3 Case 3: Organizing PacificVis

Fig. 6.4 illustrates email archives related to organizing PacificVis. It spans for more than two years, with over 2,215 messages. As can be seen from the visualization, this event started with two major flows around April, 2009, and then split into more sub-events between November, 2009 and March, 2010. By interactively having the thread displayed (Fig. 6.4 (b)), we can tell that many long email threads lie in this event.

By exploring with the contact tracks as Fig. 6.5 shows, we can find that several people have participated in PacificVis in different time periods and with different sub-events. For example, P5, P6, and P7 were involved in PacifcVis in similar time period. However, P5 was more related to the poster program (Fig. 6.5), P6 was handling the GPGPU part, (Fig. 6.6), and P7 was involved in the tutorial submission (Fig. 6.7). Some other contacts joined PacificVis event for a relatively short time (P2, P3, P4, and P8). P1 has been related to the event throughout time, but with a lower frequency of email exchange.

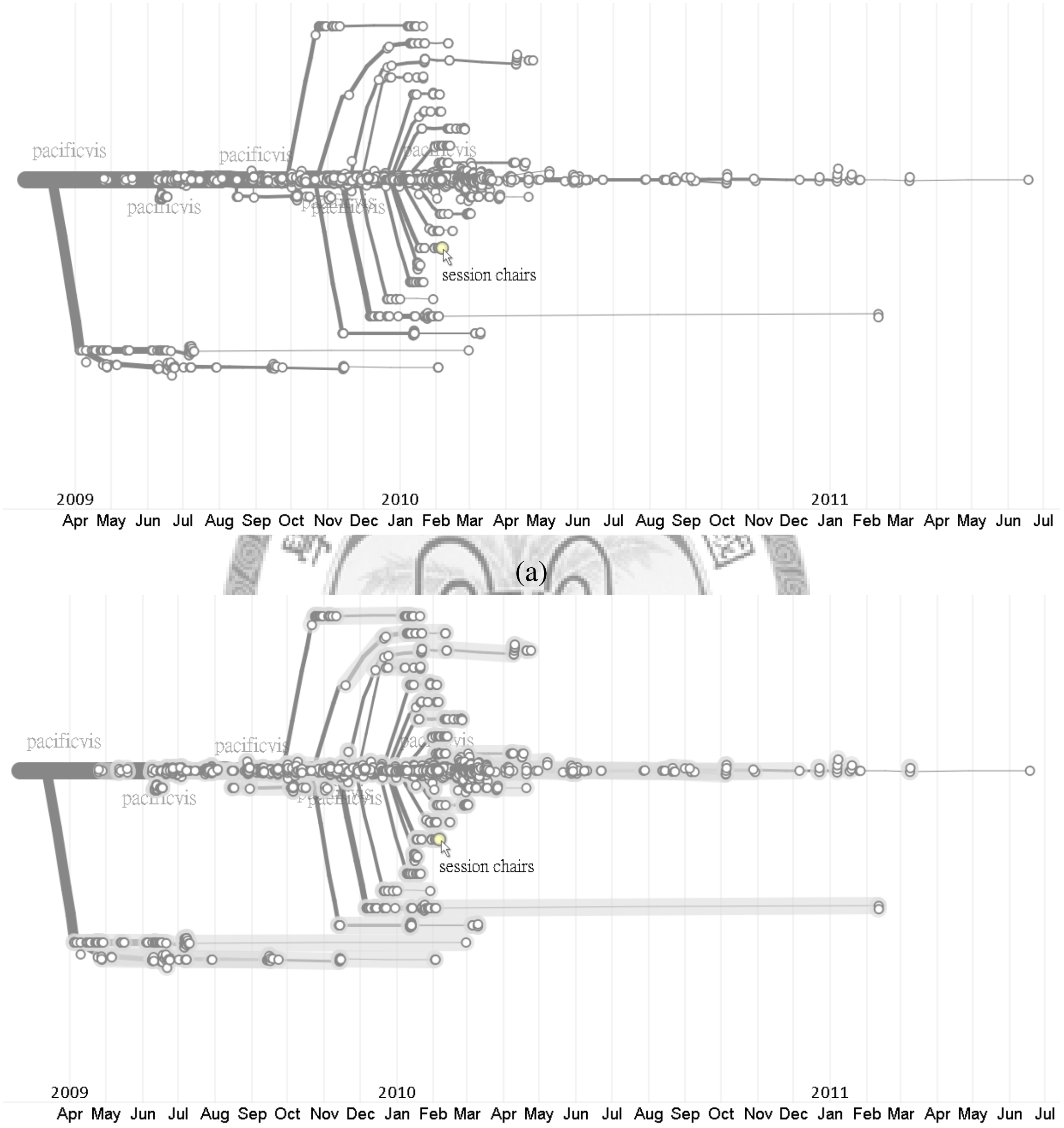
These figures provide us with how event evolved and split, and how different contacts have participated in, which can not only be used as reminiscence, but also be used for future reference. For example, the coordinators of the next year can trace the figures and get a concept of “what thing should be done around what time”.

6.2 Evaluation

As a proof-of-concept system, we conducted a small-scale user study to test and verify our assumptions. The users identified some problems, however, they also verified some functionalities of the system. We now report on the study below.

6.2.1 Methodology

In this study, we recruited six participants who were frequent email users and kept a relative long email archives. As a matter of privacy issue, we provided them a Java email parser program, and required them to download their email messages at their own com-



(a)

(b)

Figure 6.4: The visualization of large email archives with over 2,215 messages. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1000, w_{str} = 100, w_o = 1000$, and $w_t = 1000$.

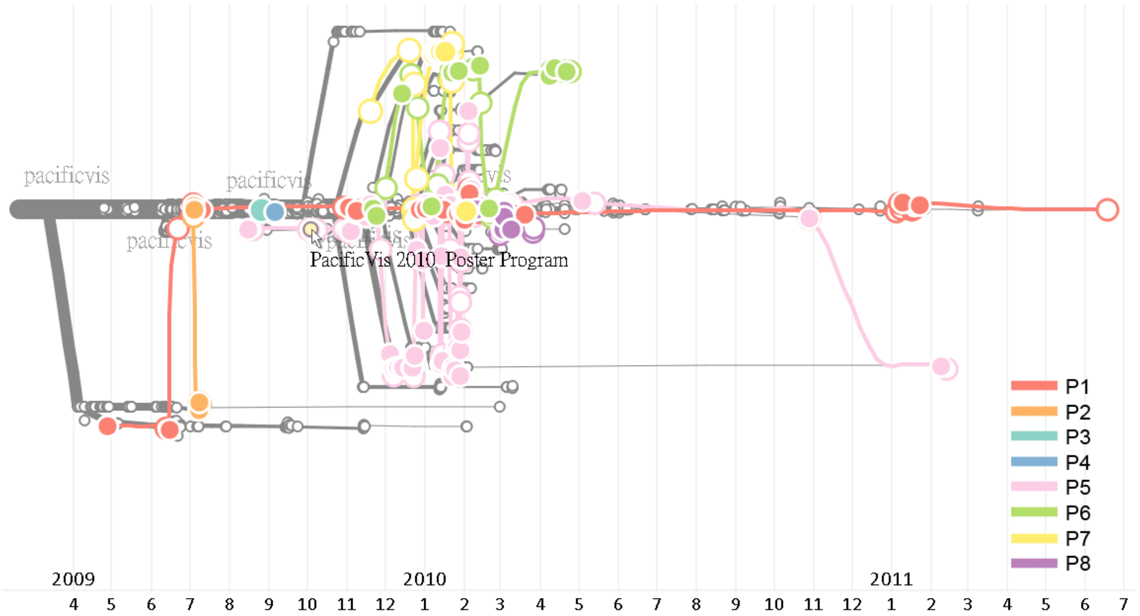


Figure 6.5: The visualization of large email archives with over 2,215 messages. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1000, w_{str} = 100, w_o = 1000$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: w_{ct_s} = 1 and $w_{ct\_d} = 500$.

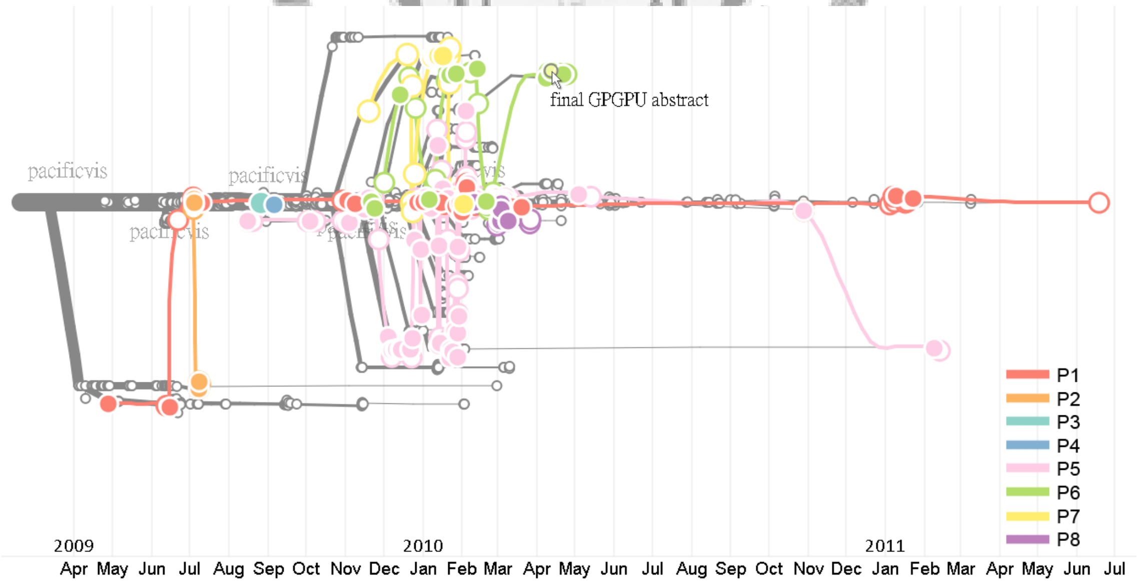


Figure 6.6: The visualization of large email archives with over 2,215 messages. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1000, w_{str} = 100, w_o = 1000$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: w_{ct_s} = 1 and $w_{ct\_d} = 500$.

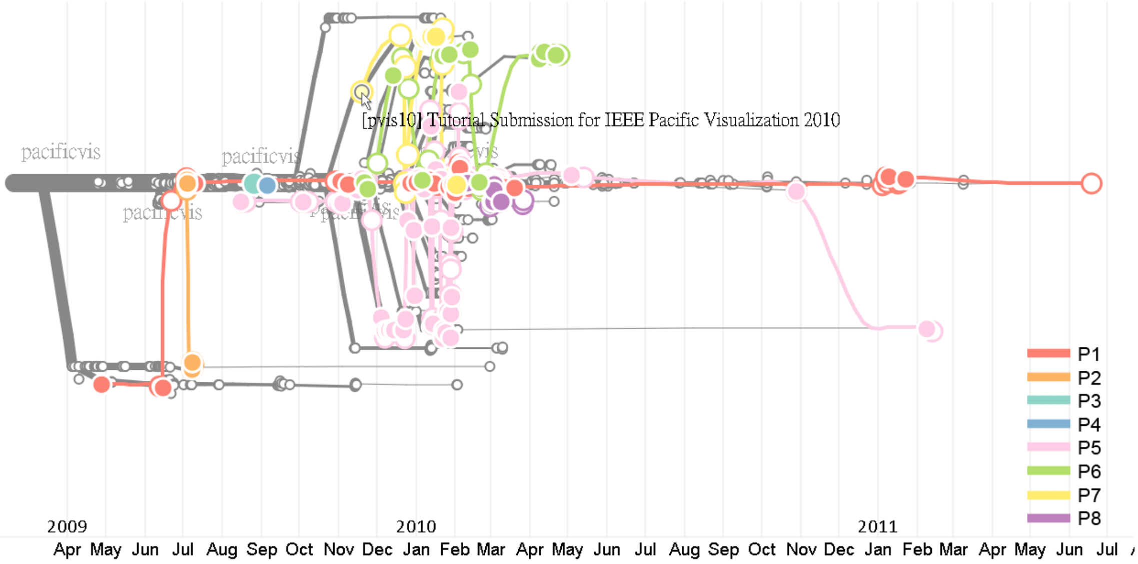


Figure 6.7: The visualization of large email archives with over 2,215 messages. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1000, w_{str} = 100, w_o = 1000$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: $w_{ct\_s} = 1$ and $w_{ct\_d} = 500$.

puter. We reminded them to checked if the resulting file contained any privacy content before providing it to us. In general, the users provided us a total of 200 to 500 email messages that were without privacy concerns. Two of the participants are female, and the participants had an average age of 25. After giving a basic instructions, users were asked to use the system freely for five to ten minutes. We required them to share any comments and feelings while using the system. The users then filled out a questionnaire evaluating EmailMap.

6.2.2 Results

In general, the users were pleased and interested when exploring their email messages in EmailMap with an average rating of 4.33 out of a 5-point Likert scale. We report on more qualitative results below.

Reminiscence. The users gave EmailMap quite positive feedbacks in terms of reminiscence. Specifically, three aspects of benefits were mentioned: to remind, to reconfirm, and to recollect.

As a tool helping people to remind, P3 mentioned: “actually, I don’t need much detail when reminiscencing. However, EmailMap provides an overview structure of the memories email have recorded. It is nice to have a high-level understanding." P4 also talked about this aspect: "The overview brings me up some memories that I don't usually think of in daily life."

P3 also talked about how EmailMap can stand as the evidence for reconfirming people's memories about the past. (P3: "Oh, the mails have proved that he used to ask me to do so much work back then.")

EmailMap also helps people to remember some events that had been forgotten. (P1: "I thought he didn't give me the answer of that homework. But he actually did!" P5: "I came to remember that we used to have a study group when exploring my email archives with the contact tracks. And there was a guy who always got blamed by the others as he didn't contribute enough.")

Structure insight. The users have verified our design purpose of providing structure insights of email archives. With both the event flow and the contact tracks, EmailMap helps the users to understand how they have interacted differently with their contacts. (P1: "I have many emails of this project at that time. Then I can see A joined this event at certain time period, which reflects that we were team members working on the final project." P4: "It is quite interesting that I can see not only how I interacted with one contact, but several of them at the same time. I can compare and see how they involved in different events." P5: "I noticed that there were lots of emails about the stock market, which continued for almost a year. I received the stock-related emails on every Monday.")

Contacts with multiple email accounts are displayed separately also revealed interesting patterns for the users. (P1: "I can see that this person has shifted from using a hotmail account to a gmail account.")

The overall structure also provided new insights that the users had not been aware of before. (P1: "Very interesting! B and a girl only intersected at those email related to going out for fun. He was into that girl, and that's why he asked her to join whenever other friends were planning to have fun.)

Orientation. Even though EmailMap was not designed for searching, one user had mention that it helped to locate the messages. (P4: "For the people I frequently contact with, it is difficult to locate a certain email message only by giving the contact's name as a search key. As I have many email messages going on with these persons, the resulting lists were always too long to look into. By using EmailMap, it is much easier to find what I was looking for with the help of context information including other contacts, other events, and the temporal patterns.)

Other feedback. One user mentioned about the clustering of emails was not good enough to help him make sense of his email archives. (P6: "I can't tell how my emails are related to each other only from the event flows. It's not well-classified enough.") As the data of P6 were mostly composed of single message without longer threads or major events detected, the relations between emails were not obvious and thus made the resulting visualization not as helpful.

The users have also commented on future possibilities of the system. For example, it can classify the contacts in terms of appearance, which would enable the user to identify the group of people that they have only contacted with once. Apart from the dimension of "event vs. time" in the current system, users were also hoping for a multi-combination views which would allow them to choose from "contact vs. time", "keywords vs. time", and so on. The experience of using EmailMap has provoked the users to imagine about the possible dimensions of information that their email archives could provide.

To conclude, the users agreed on that EmailMaps offers insights that were inaccessible with general email clients (with an average rating of 4.5 out of a 5-point Likert scale). They also found it helpful for reminiscing about past life, especially the cooperation with others, and how the cooperation changed and evolved over time (with an average rating of 4.16 out of a 5-point Likert scale).

Chapter 7

Limitation and Future Work

7.1 Limitation

While much work has been investigated on email classification [?] [?] [?] [?], it is almost impossible to construct a perfect email clustering method that can match every user's mental model of how he/she understands and classifies his/her email archives. When making sense of email archives, people might also utilize memories that lie outside of the email archives to create a richer context information that helps to better cluster the email messages.

Our similarity measure was proved being able to model people's understanding of events to a certain level. However, it might be not as precise in some cases. For example, some people might not classify their email messages purely from the email content and the people involved, but also their subjective judging of importance. An encouraging message in need could be much more meaningful than if it was sent at another time. The subjective judging of proper timing, and the personal connection in real life is difficult to measure from the email archives. When the clustering algorithm lost its preciseness, the visual representation (i.e., the event flow and the contact tracks) would be limited on depicting a meaningful story of life.

As we can see in Fig. 7.1, the thickest flow is not laid on the center but on the upper side, which indicates that most messages in the data were not grouped into significant enough events. It is likely that the possible events in the email archives were mostly short, diversified, or could only be detected by the owner's subject judging. However, while the

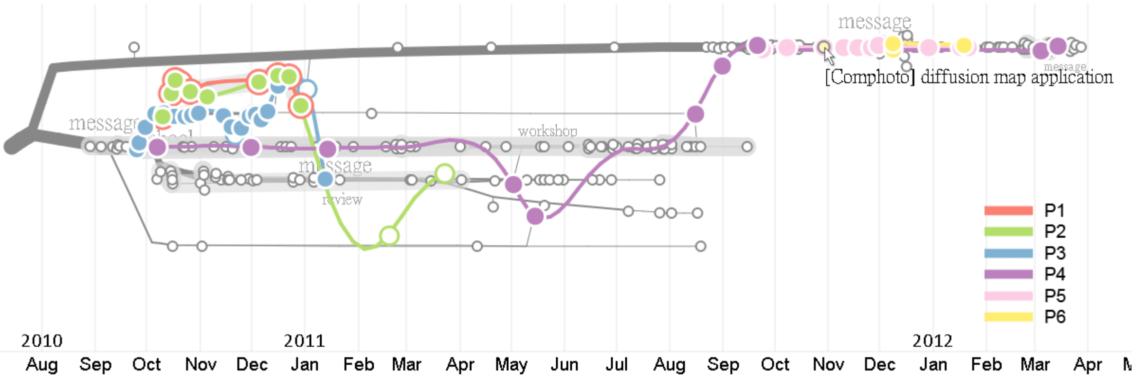


Figure 7.1: The visualization of a master student's email. Six different contact tracks are displayed. The optimization result of event flow was generated by adopting the weights as follows: $w_s = 1, w_{str} = 100, w_o = 100$, and $w_t = 1000$. The optimization result of contact tracks were generated by adopting the weights as follows: $w_{ct\_s} = 1$ and $w_{ct\_d} = 500$.

event flow was not perfectly organized, the contact tracks still depict what have happened in the past. For example, P1 and P2 have worked closely with the ego on a course project. P3 was probably the teaching assistant of another course that the ego took. P1, P2, and P3 only appeared during fall semester, 2010. P5 and P6 were related to another course in the fall semester, 2011. P4 represents the Academic Affairs Division of the email owner's university, who sent notifications to remind students of important dates from time to time.

The second limitation of this work is the detail presentation of the interaction between the contacts and ego. As we focused on and moved room for portraying an overview structure of how different people related to various life events, the detailed one-one information, such as the most frequent-talked words over time between each contact and the ego, was not shown.

7.2 Future Work

We identified the following future research directions. First, a more sophisticated email clustering algorithm is needed. There is still much need to be done on understanding how users mentally clustering their email archives into not only a flat classification, but also, an evolutional structure that can be mapped to real life experiences and memories. We are also searching for ways to integrate more details of the dyad relationship (between contacts and the ego) in the visualization, and at the same time, keeping the design intuitive and easily understandable. Other layout algorithms could also be provided to improve both the aesthetics and computational efficiency.

Chapter 8

Conclusion

As email plays a prominent role in people's communication and collaboration, it contains rich information for reminiscing and understanding of the past. However, most tools aiming on presenting email archives have limited on only one of the two aspects lying in email messages, restricting people from getting a structural comprehension of the closely related evolution of both events and the interaction between people. In this paper, we integrate the two important aspects of email archives into a single visualization. By integrating the event evolution and the interaction between people throughout time, users are enabled to make sense of their own data with complementary context information. By offering a novel approach of making sense of email archives, not only do we provide a new step for reminiscing and understanding the overall pattern of email archives, but also raise awareness for the difficult task of integrating the various information that lies in email archives.

Bibliography

[1] R. Bekkerman, A. Mccallum, and G. Huang. Automatic categorization of email into folders: Benchmark experiments on Enron and SRI corpora. Technical Report IR-418, University of Massachusetts, Amherst, 2004.

[2] R. P. Biuk-Aghai. Visualization of interactions in an online collaboration environment. In *Proceedings of the 2005 International Conference on Collaborative Technologies and Systems*, pages 228–235, 2005.

[3] K. Buchin, B. Speckmann, and K. Verbeek. Flow map layout via spiral trees. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2536–2544, Dec. 2011.

[4] V. R. Carvalho and W. W. Cohen. On the collective classification of email ”speech acts”. In *ACM SIGIR 2005 Conference Proceedings*, pages 345–352, 2005.

[5] W. Cui, S. Liu, L. Tan, C. Shi, Y. Song, Z. Gao, H. Qu, and X. Tong. TextFlow: Towards better understanding of evolving topics in text. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2412–2421, 2011.

[6] L. A. Dabbish, R. E. Kraut, S. Fussell, and S. Kiesler. Understanding email use: predicting action on a message. In *ACM CHI 2005 Conference Proceedings*, pages 691–700, 2005.

[7] J. S. Donath. Visual who: animating the affinities and activities of an electronic community. In *ACM Multimedia 1995 Conference Proceedings*, pages 99–107, 1995.

[8] M. Dork, D. Gruen, C. Williamson, and S. Carpendale. A visual backchannel for large-scale events. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1129–1138, 2010.

[9] N. Ducheneaut and V. Bellotti. E-mail as habitat: an exploration of embedded personal information management. *interactions*, 8(5):30–38, 2001.

[10] D. Fisher and P. Dourish. Social and temporal structures in everyday collaboration. In *ACM CHI 2004 Conference Proceedings*, pages 551–558, 2004.

[11] S. Frau, J. C. Roberts, and N. Boukhelifa. Dynamic coordinated email visualization. In *Proceedings of the 13th International Conference on Computer Graphics, Visualization and Computer Vision*, pages 187–193, 2005.

[12] S. Hangal, M. S. Lam, and J. Heer. MUSE: reviving memories using email archives. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, pages 75–84, 2011.

[13] S. Havre, E. Hetzler, P. Whitney, and L. Nowell. ThemeRiver: Visualizing thematic changes in large document collections. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):9–20, 2002.

[14] H. Kang, C. Plaisant, T. Elsayed, and D. W. Oard. Making sense of archived e-mail: Exploring the Enron collection with NetLens. *Journal of the American Society for Information Science and Technology*, 61(4):723–744, 2010.

[15] B. Kerr. Thread arcs: an email thread visualization. In *Proceedings of the 2003 IEEE Symposium on Information Visualization*, pages 211–218, 2003.

[16] A. Leuski. Email is a stage: discovering people roles from email archives. In *ACM SIGIR 2004 Conference Proceedings*, pages 502–503, 2004.

[17] M. Mandic and A. Kerne. Using intimacy, chronology and zooming to visualize rhythms in email experience. In *ACM CHI 2005 Extended Abstracts*, pages 1617–1620, 2005.

[18] E. Minkov, W. W. Cohen, and A. Y. Ng. Contextual search and name disambiguation in email using graphs. In *ACM SIGIR 2006 Conference Proceedings*, pages 27–34, 2006.

[19] B. A. Nardi, S. Whittaker, E. Isaacs, M. Creech, J. Johnson, and J. Hainsworth. Integrating communication and information through contactMap. *Communications of the ACM*, 45(4):89–95, 2002.

[20] C. Neustaedter, A. Brush, M. Smith, and D. Fisher. The social network and relationship finder: Social sorting for email triage. In *Proceedings of the 2nd Conference on E-mail and Anti-Spam*, 2005.

[21] A. Perer, B. Shneiderman, and D. W. Oard. Using rhythms of relationships to understand e-mail archives. *Journal of the American Society for Information Science and Technology*, 57(14):1936–1948, 2006.

[22] D. Phan, L. Xiao, R. Yeh, P. Hanrahan, and T. Winograd. Flow map layout. In *Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization*, INFOVIS '05, pages 29–, Washington, DC, USA, 2005. IEEE Computer Society.

[23] O. A. Puade and T. G. Wyeld. Visualising collaboration via email: Finding the key players. In *Proceedings of the 10th International Conference on Information Visualization*, pages 124–129, 2006.

[24] S. Rohall, D. Gruen, P. Moody, and S. Kellerman. Email visualizations to aid communications. In *Posters of the 2001 IEEE Symposium on Information Visualization*, 2001.

[25] S. Rose, S. Butner, W. Cowley, M. Gregory, and J. Walker. Describing story evolution from dynamic information streams. In *Proceedings of the 2009 IEEE Symposium on Visual Analytics Science and Technology*, pages 99–106, 2009.

[26] A. Saad and R. Dimitrios. Email threads: A comparative evaluation of textual, graphical and multimodal approaches. *International Journal of Computers*, 3(2):238–250, 2009.

[27] G. Salton, editor. *Automatic text processing*. Addison-Wesley, 1988.

[28] M. A. Smith and A. T. Fiore. Visualization components for persistent conversations. In *ACM CHI 2001 Conference Proceedings*, CHI ’01, pages 136–143, 2001.

[29] S. Sudarsky and R. Hjelsvold. Visualizing electronic mail. In *Proceedings of the 6th International Conference on Information Visualisation*, pages 3–9, 2002.

[30] J. R. Tyler and J. C. Tang. When can i expect an email response? a study of rhythms in email usage. In *Proceedings of the 8th Conference on European Conference on Computer Supported Cooperative Work*, pages 239–258, 2003.

[31] G. D. Venolia and C. Neustaedter. Understanding sequence and reply relationships within email conversations: a mixed-model visualization. In *ACM CHI 2003 Conference Proceedings*, pages 361–368, 2003.

[32] F. B. Viégas, D. Boyd, D. H. Nguyen, J. Potter, and J. Donath. Digital artifacts for remembering and storytelling: Posthistory and social network fragments. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences*, volume 4, pages 40109.1–40109.10, 2004.

[33] F. B. Viégas, S. Golder, and J. Donath. Visualizing email content: portraying relationships from conversational histories. In *ACM CHI 2006 Conference Proceedings*, pages 979–988, 2006.

[34] S. Whittaker and C. Sidner. Email overload: exploring personal information management of email. In *ACM CHI 1996 Conference Proceedings*, pages 276–283, 1996.

[35] B. Zhu and H. Chen. Communication-garden system: Visualizing a computer-mediated communication process. *Decision Support Systems*, 45(4):778–794, 2008.