Content-Centered Discussion Mapping

Ka-Ping Yee
University of California, Berkeley
ping@zesty.ca

Marti Hearst
University of California, Berkeley
hearst@sims.berkeley.edu

Abstract

We present a simple layout and visualization technique whose goal is to make the content of extended discussions more easily viewable. Unlike conversation visualizations that represent the relationships among discussion elements in a graph, we advocate emphasizing the content. The backend system processes plain text e-mail from mailing lists and extracts discussion structure based on existing formatting conventions. We iterated over several different versions of the design of the content-oriented visualization and evaluated it with a survey sent to mailing list users. Their responses were both strongly positive and strongly negative; based on these responses, we suggest a variation which may be more generally positively received in future evaluations.

Introduction and Related Work

Groups of people often engage in extended online discussions with the intention of pooling ideas and making better decisions. Unfortunately, when a discussion comes to involve many factors and viewpoints, participants can find it difficult to keep track of arguments and evidence, make corrections, and understand the rationale for an eventual resolution.

Several software projects attempt to aid the organization of shared knowledge by helping users build the content of a discussion into a predefined information structure. For example, gIBIS [3] and Compendium [2] classify discussion elements into node types such as *issue*, *position*, or *argument*, and define a fixed taxonomy of relationships by which the nodes are linked into a tree. The Coordinator [5] requires users to select one of 11 action types (such as *request*, *promise*, or *offer*) for each conversational "move." Reason!Able [6] and Tree Trellis [7] enable users to construct a tree of supporting and opposing arguments.

Some argumentation systems emphasize the use of sources and evidence, which we view as vital to making discussions productive. SenseMaker [1] lets users arrange claims into nested rectangles and place colour-coded dots representing evidence into relevant rectangles. ClaiMaker [10] lets users enter statements that paraphrase research papers and create a general graph by joining them with connectors such as "is consistent with" or "is analogous to". Rich Trellis [7] allows users to highlight fragments of arbitrary Web documents and arrange them into an analysis tree together with indicators of the perceived reliability of each source.

Many different visualization techniques have been proposed for threaded discussions, such as Kerr's Thread Arcs [9], Venolia and Neustaedter's mixed-model visualization [12], and Smith and Fiore's thread trees [11]. Engdahl et al. have also suggested a treemap [4] for visualizing threaded conversations on the limited screen space of a PDA.

Approach

While there is much value to be gained from a structured representation, we feel that too much emphasis on the classification of arguments and their relationships impedes flexibility and usability in practice. ClaiMaker and Rich Trellis offer dozens of logical connectors, focusing on annotating and formalizing existing documents in order to enable automated reasoning, whereas we are interested in helping the users understand each other and find consensus.

Many designs implicitly assume that users will follow the rules of the system and employ the components of the system as the designers intended. Though it may be possible to expect such conformance in an educational setting, in general one cannot even assume that users will choose to use the system at all. Therefore, our primary assumption is that participants will continue to use their current tools and abide by current practices unless they have a compelling motivation to do otherwise. We aim to facilitate rather than to prescribe constructive behaviour.

E-mail Processing

Electronic mailing lists are an extremely popular tool for long-term discussion. They are easy to understand, straightforward to administer, and require no special client software. Since mailing lists are such a prevalent discussion medium, we have chosen them as the baseline to augment.

Our tool processes messages as they arrive, generating and updating a Web-based display to help organize arguments and evidence. Participants can guide the construction of this display by following the formatting conventions we describe below, but they are not required to do so. Thus, as they write messages they are also contributing to a shared knowledge artifact. We aim to provide a display useful enough that participants want to use it in addition to their e-mail client. (In future we may add Web-based posting.)

The most common method of arranging e-mail messages into threads is to make a tree where the nodes are entire messages and child nodes represent replies. We can reveal more of the content in each message by taking advantage of a common convention: users often quote relevant lines of another message, prefix the lines of the quotation with ">", and follow the quotation with their reply. In Zest [13] and in our current design, each instance of this construct yields a new reply node linked to its parent. Multiple reply nodes can share the same parent but quote different parts of it.

A message that does not reply to any others introduces a new topic at the root of a new tree. A replying node is classified as a supporting or opposing statement if the text begins with "`[+]`" or "`[-]`". (A participant can reply with one of these marks and no text to merely indicate support or opposition.) A replying node is classified as a question if the text begins with a sentence ending in a question mark.

First Design Iteration

Figure 1 shows the initial design of our visualization. The text block running across the top of the figure is the focal topic of the thread. Positions and arguments on the topic are placed in blocks from left to right below the topic block. Further replies are in turn arranged from left to right beneath their parent blocks, with supporting blocks on the left and opposing blocks on the right. Questions are shown in smaller orange blocks nested within their parent blocks.

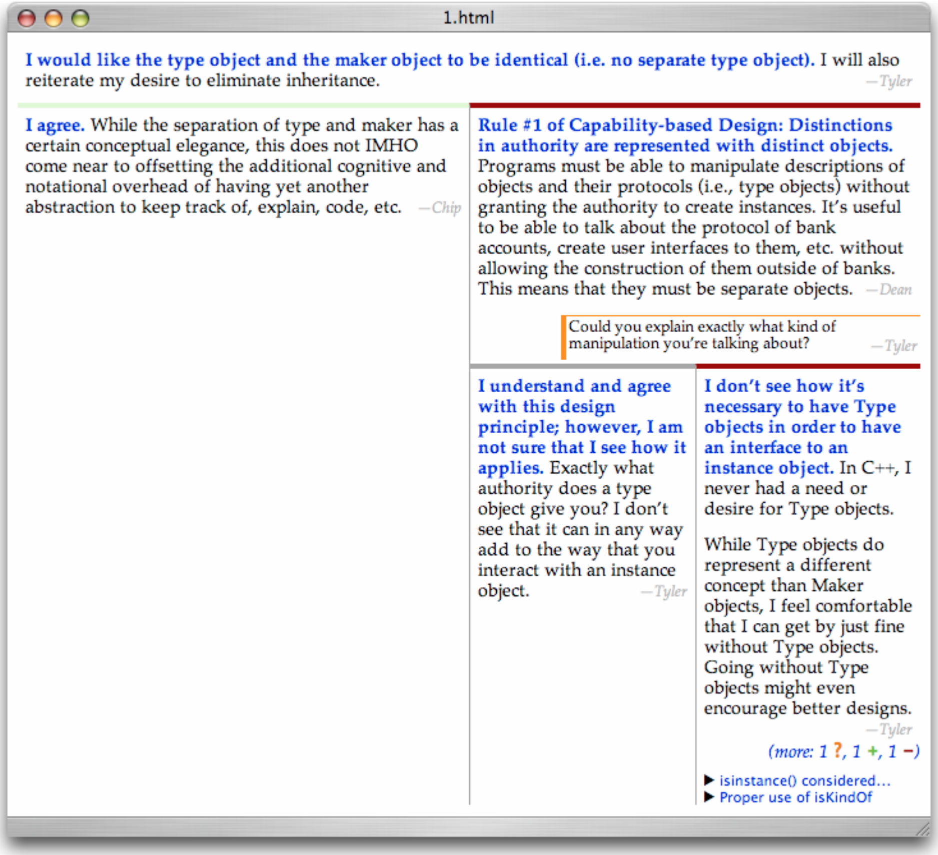
This layout technique (reminiscent of a treemap [8]) is designed to address two problems with the typical outline-style layout of a threaded conversation. First, when shown using lists indented within lists, the nodes of a conversation tree appear in depth-first order, which places sibling nodes vertically far apart: replies to earlier siblings push later siblings further away from their parent. Placing responses in columns brings them nearer to their parents and gives them more equal footing; none of them can be pushed off the bottom of the screen. A second problem with indented lists is that they waste space. Even when the conversation is linear, with each node replying to the last, successive levels are indented further and further. In our layout, a linear conversation becomes a single column of text blocks.

Our layout method has the drawback of limiting the depth and breadth of the tree because narrow columns of text are hard to read. Our current solution is to show only a few levels and allow the user to click on a node to navigate to deeper levels of detail. When a non-root node is the current focus, as in Figure 2, the first sentences of its ancestors are shown in small type above the focus node to give context.

We expect that references to sources and supporting evidence will be cited as URLs, so URLs found in the text are called out and displayed first for emphasis. All the URLs in nodes too deep to be displayed are listed in the nearest visible node, which makes citations easier to find and also makes the absence of citations more obvious.

Questions are pulled out and separated from the main tree of the discussion in order to distinguish the main arguments from clarifications and to allow the flagging of unanswered questions to help promote their resolution. Once a participant has answered a question concerning one of his or her own statements, anyone may quote an excerpt of the answer and indicate their satisfaction with a "`[+]`" mark, which causes the selected excerpt to appear next to the original statement where it can serve as a clarification or correction.

The design strives to encourage certain behaviours. The first sentence of each block is shown in bold, encouraging users to write paragraphs that begin with topic sentences. Author names are de-emphasized in order to focus attention on substance rather than speakers. Though it may seem simplistic to assume every node beginning with a question is a request for clarification, we are interested in finding out if this rule will encourage participants to make arguments using direct statements instead of rhetorical questions.



I would like the type object and the maker object to be identical (i.e. no separate type object). I will also reiterate my desire to eliminate inheritance. —Tyler

I agree. While the separation of type and maker has a certain conceptual elegance, this does not IMHO come near to offsetting the additional cognitive and notational overhead of having yet another abstraction to keep track of, explain, code, etc. —Chip

Rule #1 of Capability-based Design: Distinctions in authority are represented with distinct objects. Programs must be able to manipulate descriptions of objects and their protocols (i.e., type objects) without granting the authority to create instances. It's useful to be able to talk about the protocol of bank accounts, create user interfaces to them, etc. without allowing the construction of them outside of banks. This means that they must be separate objects. —Dean

Could you explain exactly what kind of manipulation you're talking about? —Tyler

I understand and agree with this design principle; however, I am not sure that I see how it applies. Exactly what authority does a type object give you? I don't see that it can in any way add to the way that you interact with an instance object. —Tyler

I don't see how it's necessary to have Type objects in order to have an interface to an instance object. In C++, I never had a need or desire for Type objects.

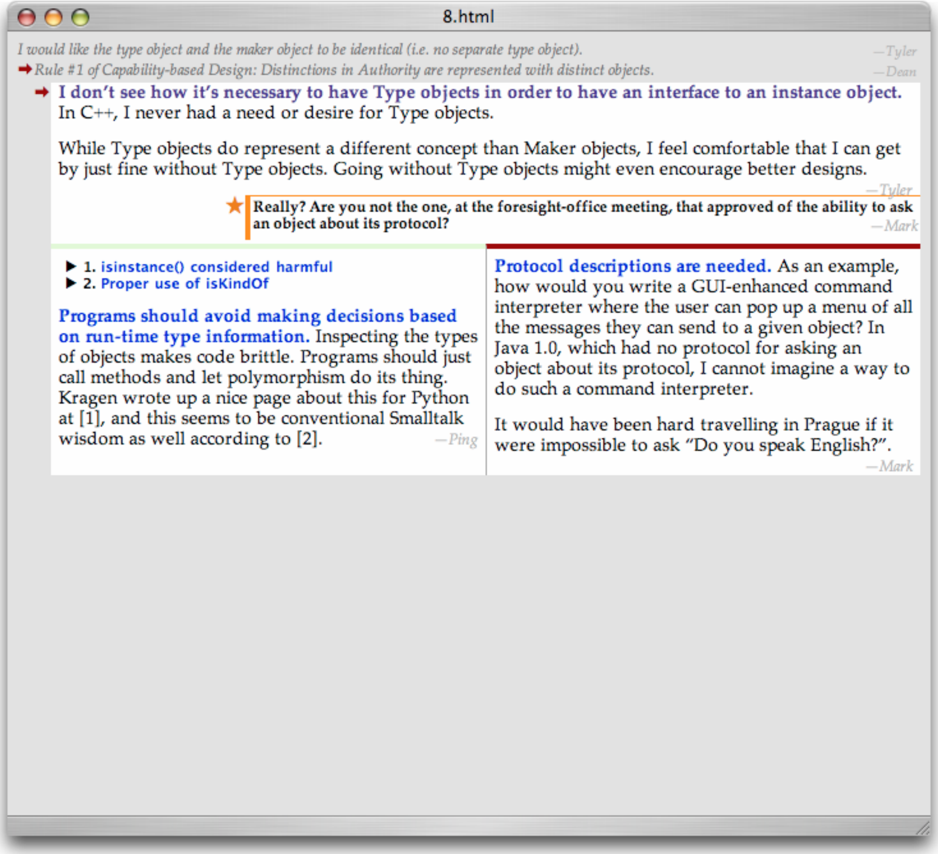
While Type objects do represent a different concept than Maker objects, I feel comfortable that I can get by just fine without Type objects. Going without Type objects might even encourage better designs. —Tyler

(more: 1 ?, 1 +, 1 −)

▶ isinstance() considered…
▶ Proper use of isKindOf

Figure 1. In this view, the focus is on the root node of a topic, shown at the top of the display. The space below the root node is divided between a supporting statement on the left and an opposing statement on the right. The left block has a light green bar on top to show agreement with the root node, while the right block has a dark red bar on top to suggest opposition.

One question, shown in a nested block with an orange border, has been asked in response to the statement beginning "Rule #1". Two replies to this statement appear below it, one neutral and one opposing. Supporting and opposing blocks come from text sections that were prefixed with [+] and [-] in the original messages; the neutral block comes from an unmarked paragraph.



I would like the type object and the maker object to be identical (i.e. no separate type object). —Tyler

➡ *Rule #1 of Capability-based Design: Distinctions in Authority are represented with distinct objects.* —Dean

➡ **I don't see how it's necessary to have Type objects in order to have an interface to an instance object.** In C++, I never had a need or desire for Type objects.

While Type objects do represent a different concept than Maker objects, I feel comfortable that I can get by just fine without Type objects. Going without Type objects might even encourage better designs. —Tyler

★ **Really? Are you not the one, at the foresight-office meeting, that approved of the ability to ask an object about its protocol?** —Mark

▶ 1. isinstance() considered harmful
▶ 2. Proper use of isKindOf

Programs should avoid making decisions based on run-time type information. Inspecting the types of objects makes code brittle. Programs should just call methods and let polymorphism do its thing. Kragen wrote up a nice page about this for Python at [1], and this seems to be conventional Smalltalk wisdom as well according to [2]. —Ping

Protocol descriptions are needed. As an example, how would you write a GUI-enhanced command interpreter where the user can pop up a menu of all the messages they can send to a given object? In Java 1.0, which had no protocol for asking an object about its protocol, I cannot imagine a way to do such a command interpreter.

It would have been hard travelling in Prague if it were impossible to ask "Do you speak English?". —Mark

Figure 2. Clicking on the "more" link or the first sentence of the bottom-right block in Figure 1 shifts the focus to the node beginning with "I don't see…" , yielding the display shown here. The two ancestors are shown on single lines above the focus node. The question in this figure is marked with a star to indicate that it remains unanswered.

The supporting block on the lower left was generated by a message that cited two URLs. These URLs are replaced with the numbers "[1]" and "[2]" in the text and listed as hyperlinks at the beginning of the block. The text of each link comes from the title of the referenced webpage. The lower-right block in Figure 1 also shows these links in its list of all URLs in its descendants.

Second and Third Design Iterations

To test the design concept, we ran a preliminary survey of mailing list and newsgroup participants. Since it was infeasible to force an entire group of participants to use the visualization for an extended period of time, our questionnaire just asked participants to provide their feedback on a visualization of a discussion that had already taken place.

The survey was conducted using a second iteration of the design (not shown) that was aimed more specifically at newsgroup and e-mail practices and did not rely on text markers such as "`[+]`", "`[-]`", or "?", since the visualization was generated from previously written public messages. The second iteration improved on the original design by shading the background of each block in a colour corresponding to its author. The colours of the blocks identify authors who tend to write frequently or infrequently, and help reveal patterns in behaviour such as alternation between two participants. Also, instead of simply truncating the message tree at a certain depth, the second design shows a collapsed view of the deeper part of the tree using coloured blocks without text. This gives a clear visual indication of the amount and type of activity further down the thread.

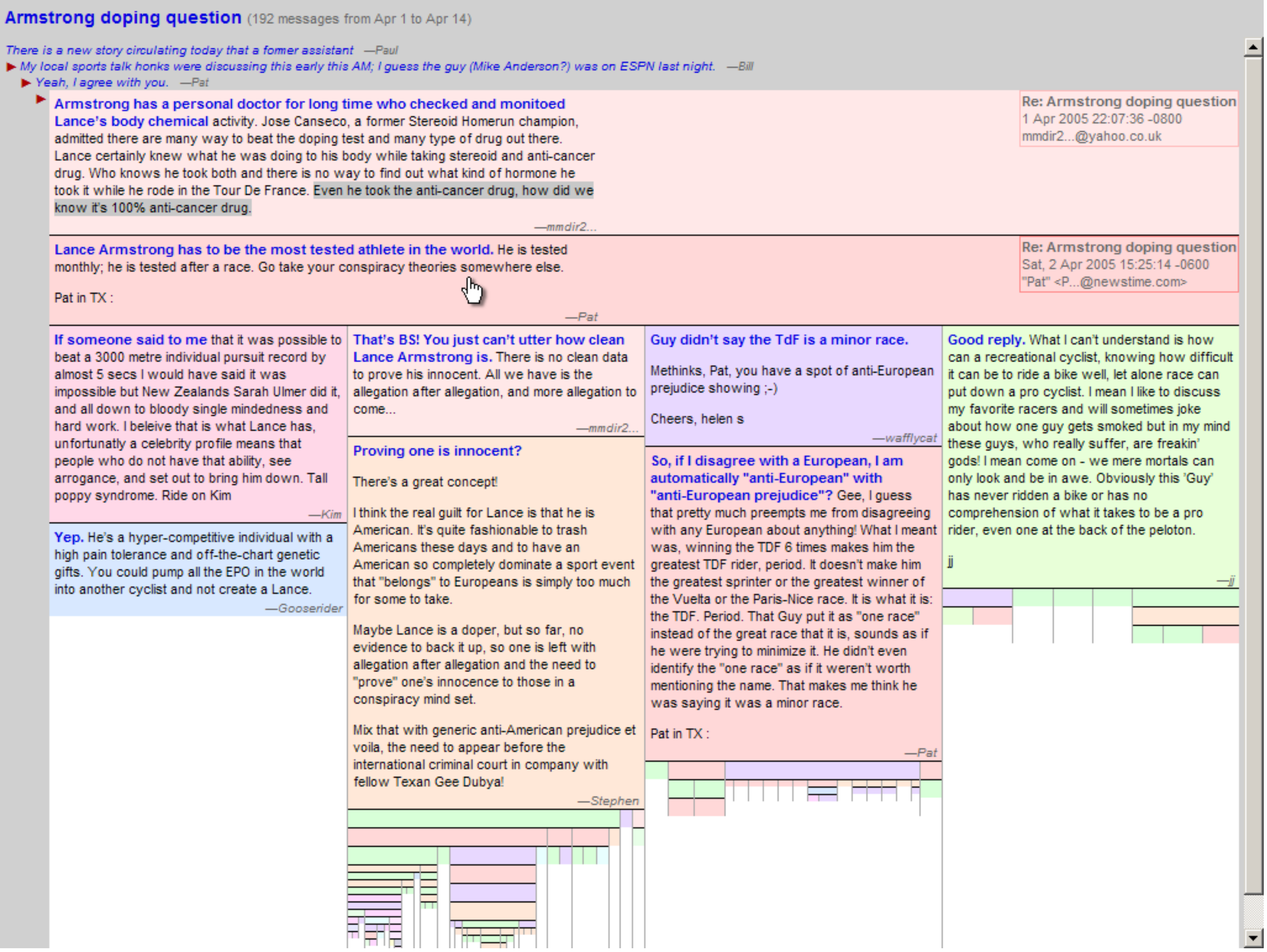
Based on feedback from the survey, we made other minor improvements in a third design iteration, shown in Figure 3. We incorporated feedback from respondents who wanted to see more of the message header information, particularly the original date and the e-mail address of the author. To prevent lines of text from becoming too long and hard to read, we limited the maximum width of text in wide blocks, such as the top two blocks in Figure 3. When the mouse cursor is positioned over a message block that quotes part of its parent, the quoted part is highlighted in the parent block. Finally, in the third iteration, the user can drill down by clicking anywhere in a child block rather than having to click specifically on the first sentence.

Survey Questions

We selected three active Usenet newsgroups (`rec.juggling`, `rec.bicycles.misc`, and `rec.pets.cats.health+behav`) and two mailing lists (developer lists for the Python programming language and the Scribus desktop publishing program). In each group, we produced a visualization of a recent conversation thread and posted a message on the group inviting members to participate in our online survey. The questionnaire first asked participants to indicate:

- the length of their membership in the particular discussion group
- their experience using online discussion groups of any kind
- the program or service used to read new messages and review past messages
- the frequency of reading new messages and reviewing past messages
- the purpose for participation in the group

Then, we presented the visualization and asked the participants to evaluate it along several different criteria, such as how well it could be understood, whether it would be preferred over the current method of reading messages, and so on. These factors are detailed below. Finally, participants were encouraged to write free-form comments.



Armstrong doping question (192 messages from Apr 1 to Apr 14)

There is a new story circulating today that a fomer assistant —Paul

▶ *My local sports talk honks were discussing this early this AM; I guess the guy (Mike Anderson?) was on ESPN last night.* —Bill

▶ *Yeah, I agree with you.* —Pat

▶ **Armstrong has a personal doctor for long time who checked and monitoed Lance's body chemical** activity. Jose Canseco, a former Stereoid Homerun champion, admitted there are many way to beat the doping test and many type of drug out there. Lance certainly knew what he was doing to his body while taking stereoid and anti-cancer drug. Who knows he took both and there is no way to find out what kind of hormone he took it while he rode in the Tour De France. Even he took the anti-cancer drug, how did we know it's 100% anti-cancer drug.

—mmdir2...

Re: Armstrong doping question
1 Apr 2005 22:07:36 -0800
mmdir2...@yahoo.co.uk

Lance Armstrong has to be the most tested athlete in the world. He is tested monthly; he is tested after a race. Go take your conspiracy theories somewhere else.

Pat in TX :

—Pat

Re: Armstrong doping question
Sat, 2 Apr 2005 15:25:14 -0600
"Pat" <P...@newstime.com>

If someone said to me that it was possible to beat a 3000 metre individual pursuit record by almost 5 secs I would have said it was impossible but New Zealands Sarah Ulmer did it, and all down to bloody single mindedness and hard work. I beleive that is what Lance has, unfortunatly a celebrity profile means that people who do not have that ability, see arrogance, and set out to bring him down. Tall poppy syndrome. Ride on Kim

—Kim

Yep. He's a hyper-competitive individual with a high pain tolerance and off-the-chart genetic gifts. You could pump all the EPO in the world into another cyclist and not create a Lance.

—Gooserider

That's BS! You just can't utter how clean Lance Armstrong is. There is no clean data to prove his innocent. All we have is the allegation after allegation, and more allegation to come...

—mmdir2...

Proving one is innocent?

There's a great concept!

I think the real guilt for Lance is that he is American. It's quite fashionable to trash Americans these days and to have an American so completely dominate a sport event that "belongs" to Europeans is simply too much for some to take.

Maybe Lance is a doper, but so far, no evidence to back it up, so one is left with allegation after allegation and the need to "prove" one's innocence to those in a conspiracy mind set.

Mix that with generic anti-American prejudice et voila, the need to appear before the international criminal court in company with fellow Texan Gee Dubya!

—Stephen

Guy didn't say the TdF is a minor race.

Methinks, Pat, you have a spot of anti-European prejudice showing ;-)

Cheers, helen s

—wafflycat

So, if I disagree with a European, I am automatically "anti-European" with "anti-European prejudice"? Gee, I guess that pretty much preempts me from disagreeing with any European about anything! What I meant was, winning the TDF 6 times makes him the greatest TDF rider, period. It doesn't make him the greatest sprinter or the greatest winner of the Vuelta or the Paris-Nice race. It is what it is: the TDF. Period. That Guy put it as "one race" instead of the great race that it is, sounds as if he were trying to minimize it. He didn't even identify the "one race" as if it weren't worth mentioning the name. That makes me think he was saying it was a minor race.

Pat in TX :

—Pat

Good reply. What I can't understand is how can a recreational cyclist, knowing how difficult it can be to ride a bike well, let alone race can put down a pro cyclist. I mean I like to discuss my favorite racers and will sometimes joke about how one guy gets smoked but in my mind these guys, who really suffer, are freakin' gods! I mean come on - we mere mortals can only look and be in awe. Obviously this 'Guy' has never ridden a bike or has no comprehension of what it takes to be a pro rider, even one at the back of the peloton.

jj

—jj

Figure 3. In this thread, newsgroup participants on **rec.bicycles.misc** debate the validity of claims that Lance Armstrong used performance-enhancing drugs. Message blocks written by the same author are shaded in the same colour. The mouse cursor is pointing at the second block, which begins "Lance Armstrong has to be the most tested athlete in the world." The sentence beginning "Even he took…" is highlighted in the first block to indicate that the author quoted that sentence immediately before writing the sentence in the second block.

Survey Results

We received 12 survey responses from `rec.pets.cats.health+behav`, 35 responses from `rec.juggling`, 204 responses from the Python mailing list, and 16 responses from the Scribus mailing list.

Respondents had widely diverging opinions about the visualization. Some called it "unsightly," "a horrible coloured mess," "jumbled and confusing," or "far too cluttered and complex," while others described it as "beautiful," "awesome," or "brilliant." Here are some of the general comments (in which "threadmap" refers to our visualization):

- "The threadmap's advantages became apparent after choosing a deeply nested node of the thread. The display showing that message, its predecessors and the subsequent branching of the topic from that point is remarkable."
- "I was surprised at how effectively the threadmap captured the flow of the conversation."
- "The threadmap is far superior to web-based mailing list archives."
- "Very very nice. I like the way it presents the hierarchy of threads — much easier to scan than a treeview. Easy to see which threads are hot and which are dead."
- "Didn't like it at all. I prefer to read a message in its entirety and then move on to the next message."
- "While I appreciat [sic] what you are trying to do, I found your method much too confusing and more dificult [sic] to follow. Back to the drawing board!"
- "It looks like a useful tool to visualise a complete thread. Unfortunately I generally don't want to visualise a thread, I want to read it."
- "The idea is nice, but the web is not a good place to implement it. The interface is not very nice. IMO it would have to be much more dynamic with collapsing and expanding at will without the delay from loading the page."
- "Shifting back and forth reading down and sideways drove me nuts."
- "I'm really impressed by the way you made it somewhat natural to read — the reading order and layout is sensible for reading through the thread in a linear way. That's nice."

Among the free-form comments, respondents identified some specific concerns:

- 10 respondents wrote comments that showed they had trouble understanding the meaning of the layout.
- 9 respondents expressed dislike or concerns about the horizontal layout of child nodes; some were concerned that the display would become too wide when a message had too many replies.
- 9 respondents called attention to the lack of a text search feature.
- 6 respondents wanted more emphasis on the names of authors.
- 5 respondents were concerned about messages being split up into pieces.
- 4 respondents called attention to the lack of filtering or sorting features.
- 4 respondents wanted to be able to keep track of read and unread messages.

Evaluation Criteria

The following table shows how respondents evaluated our visualization according to several specific criteria. Note that one of the authors is an active member of the Python mailing list, and responses from that group were generally much more positive than the rest, so those responses have been excluded from this table.

	strongly disagree	disagree	neutral	agree	strongly agree
The threadmap helps me understand the conversation.	13	16	10	23	0
The threadmap is easy to navigate.	13	14	8	24	3
The emphasis on the first sentence in each block helps me understand the conversation.	8	14	10	16	2
The highlighting of sentences in the parent block helps me understand the conversation.	8	12	18	11	0
After spending some time to become familiar with the threadmap, I would be comfortable using it to browse conversations.	6	12	7	20	4
For reading a currently ongoing conversation, I would prefer to use the threadmap over my current method.	26	13	9	12	1
For reviewing a past conversation, I would prefer to use the threadmap over my current method.	21	9	11	17	3
I would use the threadmap in addition to my current method.	11	10	7	17	3
Using the threadmap is faster than my current method for figuring out what is happening in a conversation.	13	8	7	15	6
If the threadmap were always available, I would probably look at past conversations more often.	19	17	16	6	1

Conclusions and Future Work

It is clear from the responses to the survey that, although it has a few enthusiastic fans, our design is not ready for general use. The comments suggest several possible variations on the design, particularly a transposed layout such as the one sketched in Figure 4, where the child nodes are displayed in columns to the right of their parents Displaying the nodes in columns has the potential to address several problems at once: the columns would have fixed widths, so lines of text would not become too short or too long; child nodes would be ordered vertically, so consecutive parts of the same message would not be separated; and the required eye movement would be simpler, travelling simply in columns from left to right rather than among blocks of all different dimensions.

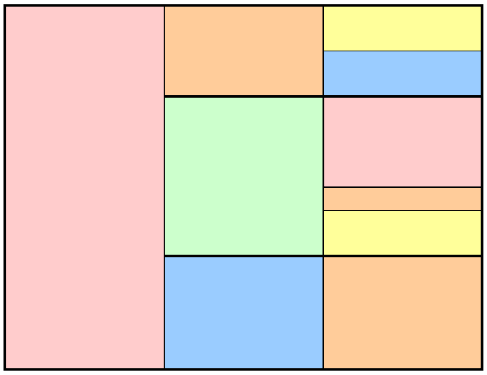


Figure 4. A transposed, column-based variation on our visualization design. Children are to the right of their parents.

We feel that this design space offers many interesting possibilities for further exploration. The colouring of blocks can be used to indicate a message's rating, age, or read wear instead of its author. The distribution of horizontal or vertical space among child nodes could be adjusted to indicate the relative levels of support for statements. A search interface is clearly a necessary feature to help participants locate relevant arguments from previous discussions; a possible design would be to provide a search field that filters the blocks immediately as each character is typed, highlighting the blocks that contain the search string. As one respondent noticed, it may be helpful to provide smoothly animated graphical transitions as the user navigates up and down the thread. For example, a zooming user interface is a natural fit for navigating a treemap.

The survey we conducted only evaluates the visualization as a way of viewing a previous conversation; it tells us little about the experience of using such a visualization tool while participating in the discussion. Our designs still need to be tested with a group of participants who all use them on an ongoing basis while discussing new topics.

Acknowledgements

This work was supported by an IBM Ph. D. Fellowship. Thanks also to Jeffrey Heer and Andrew T. Fiore for their suggestions and feedback.

References

1. Philip Bell. Using Argument Representations to Make Thinking Visible for Individuals and Groups. Proceedings of CSCL 1997. http://www.oise.utoronto.ca/cscl/papers/bell.pdf.
2. Compendium Institute. http://compendiuminstitute.org/.
3. Jeff Conklin, Michael Begeman. gIBIS: A Tool for Exploratory Policy Discussion. Proceedings of CSCW 1988.
4. Björn Engdahl, Malin Köksal, Gary Marsden. Using Treemaps to Visualize Threaded Discussion Forums on PDAs. Extended Abstracts of ACM CHI 2005.
5. Fernando Flores, Michael Graves, Brad Hartfield, Terry Winograd. Computer Systems and the Design of Organizational Interaction. ACM Transactions on Office Information Systems 6(2), April 1988.
6. Tim van Gelder. Learning to Reason: A Reason-Able approach. Proceedings of the Fifth Australasian Cognitive Science Society Conference. http://philosophy.unimelb.edu.au/reason/papers/AustCogSci2000.pdf.
7. Yolanda Gil, Varun Ratnakar. Trusting Information Sources One Citizen at a Time. Proceedings of the First International Semantic Web Conference. http://trellis.semanticweb.org/expect/web/semanticweb/iswc02_trellis.pdf.
8. Brian Johnson, Ben Shneiderman. Tree-Maps: a Space-Filling Approach to the Visualization of Hierarchical Information Structures. Proceedings of IEEE Visualization 1991.
9. Bernard Kerr. Thread Arcs: An Email Thread Visualization. Proceedings of IEEE Information Visualization 2003.
10. Gangmin Li, Victoria Uren, Enrico Motta, Simon Buckingham Shum, John Domingue. ClaiMaker: Weaving a Semantic Web of Research Papers. http://kmi.open.ac.uk/publications/papers/kmi-tr-126.pdf.
11. Marc A. Smith, Andrew T. Fiore. Visualization Components for Persistent Conversations. Proceedings of ACM CHI 2001.
12. Gina D. Venolia, Carman Neustaedter. Understanding Sequence and Reply Relationships within Email Conversations: A Mixed-Model Visualization. Proceedings of ACM CHI 2003.
13. Ka-Ping Yee. Zest: Discussion Mapping for Mailing Lists. Extended Abstracts of CSCW 2002. http://zesty.ca/pubs/cscw-2002-zest.pdf.