Showtime: Increasing viewer understanding of dynamic network visualisations

Ross Shannon
Systems Research Group,
School of Computer Science and Informatics,
UCD Dublin, Ireland
ross.shannon@ucd.ie

Aaron Quigley
HITLab Australia,
School of Computing and Information Systems,
University of Tasmania, Australia
aaron.quigley@utas.edu.au

Paddy Nixon
Systems Research Group,
School of Computer Science and Informatics,
UCD Dublin, Ireland
paddy.nixon@ucd.ie

ABSTRACT
Visualisations of dynamic networks are animated over time, reflecting changes in the underlying data structure. As viewers of these visualisations, it is up to us to accurately perceive and keep up with the constantly shifting view, mentally noting as visual elements are added, removed, changed and rearranged, sometimes at great pace. In a complex data set with a lot happening, this can put a strain on the observer's perceptions, with changes in layout and visual population disrupting their internalised mental model of the visualisation, making it difficult to understand what the changes represent. We present *Showtime*, a novel visualisation technique which dilates the flow of time so that observers have proportionally more time to understand each change based on the density of activity in the visualisation. This is paired with a novel timeline element which tracks the flow of time visually.

Categories and Subject Descriptors
H.1.2 [**Models and Principles**]: User/Machine Systems—*Human factors, human information processing;*
H.5.2 [**Information Interfaces And Presentation (e.g., HCI)**]: User Interfaces—*theory and methods*.

General Terms
Design, Human Factors.

Keywords
Information visualization, time-series, graphs, dynamic graph drawing, temporal manipulation.

1. INTRODUCTION
Most traditional visualisation techniques have been developed with *static* data in mind. When developing visual tools to analyse these data, the scope of the entire data set is known to the developer, and the view generated from this data is often static too: a drawing of a large graph or scatterplot which could then be analysed and pored over at the observer's leisure to find patterns, identify outlying cases and hypothesise on causality [16].

Other data sources are *dynamic*: they change over time. Examples abound, from the evolution of a person's social network throughout their life to the more transitory device populations of ad-hoc networks. These types of data present the visualisation designer with options: they can take a snapshot of the data as it existed at a certain time, generate a static view from it and perform a visual analysis of its structure and layout, as before. This lets a user analyse aspects of the data, but doesn't allow them to see the data evolve over time.

Further understanding can be extracted by continually visualising the data at successive moments in its evolution, and transitioning smoothly between instants using animation. The view an observer sees at any particular time step is thus only a view of the system for that instant; the view may be modified or entirely replaced as the visualisation progresses. This kind of visualisation presents a different variety of information about the data: at what rate and regularity do changes occur, how does the network under study respond to changes in topology, and where do localised changes occur within the overall structure? The visualisation observer watches as the data evolves before them, unravelling the structure and history of the information.

For instance, an ad-hoc network of wireless mobile devices will evolve over time as new devices come into range and join the network. Connections between the devices will also be transient. Thus, a visualisation of this data set will need to show nodes entering and leaving, edges being instantiated and deleted, and so on [12], all occurring at irregular intervals on different timescales. Similarly a visualisation of a person's social network will grow in complexity over long periods of time, when further nodes join the network and rich interconnectivity emerges.

Here, we present *Showtime*, an approach to harnessing the user's ability to understand change when presented at the right pace by controlling the flow of time within the visualisation. This technique gives the human observer more time to absorb and assimilate changes into their understanding of the display, while also giving the visualisation algorithm more time to manage and represent these changes on-screen. The next section describes the kinds of change that are in effect in dynamic network visualisations. Following this, we introduce our model of *event sequences,* and show how this technique adds a rigour to an observer's naturally imprecise conception of time.

2. DYNAMIC VISUALISATIONS
Sources of dynamic data exist in many fields, and depending on the speed with which the data in question changes and is recorded, have a range of tempos and rhythms. In economics, the noise-

AVI '10, May 25-29, 2010, Rome, Italy

laden fluctuations of the stock market generate huge amounts of data each day the markets open and are frequently analysed in real-time to support critical quick decisions. In contrast, the rapidly-changing protein interactions or metabolic networks that are analysed in bio-informatics are more suitable for deeper analysis after the fact. On the other end of the scale, the evolution of a social network happens at a comparatively slow speed, but can be sped up to be visualised and analysed at a pace more befitting visual inspection [3].

In this research, we investigate network visualisations, which can range from the sociograms of social science to the visualisation of telecommunications networks large and small. Graph drawings can be judged over a number of aesthetic criteria—such as the number of visible edge crossings, or the aspect ratio of the diagram—which have an effect on a user's understanding of the structure and features of the network [11].

Dynamic graph drawing is a separate branch of research from work on the traditional static plots, and introduces techniques to redraw the graph after a change has occurred [1]. In these type of dynamic networks, new information is added, while old information is refined or replaced. When these changes occur, it is important to minimise disruption as much as possible [4]. The changes that can occur in a graph include:

- nodes being added or removed
- edges being added or removed
- changes in the weight of either nodes or edges
- changes in the explicit clustering of nodes

Some of these change events can be seen in Figure 1. From these images, you may be able to see that it can be difficult to identify where exactly the graph has changed, as well as knowing the full extent of the change. Bear in mind that you have the benefit of being able to compare the views of the graph side-by-side to help identify differences (what Tufte calls *small multiples* [15]), as well as an explanatory caption. In an animation of a dynamic graph with nodes appearing and disappearing all the time without warning, these kind of comparisons are not possible, and the observer is forced to compare the views in *time*, rather than in *space*, relying on their memory and observational skills to discern the differences from frame to frame rather than purely leveraging their visual perception.

2.1. The User's Mental Model

A visualisation in flux creates a host of problems for the viewer. While analysing any graph visualisation, dynamic or not, a user builds up what's called a *mental model* [2]. Visible changes to the graph are incorporated by the user into their own mental model as they are perceived, and many techniques have been established for minimising the visual disruption caused by a change to the graph~ [6]. When the mental model *is* disrupted, the user may struggle to reconcile a change with their understanding of the structure of the data, and lose their built-up understanding, forcing them in the worst case to have to begin again with learning the structure of the network, while new modifications continue to be made.

Multiple fleeting changes can effectively co-occur in the view, in different parts of the screen, which increases the possibility of the viewer missing vital changes. The irregularity of changes also

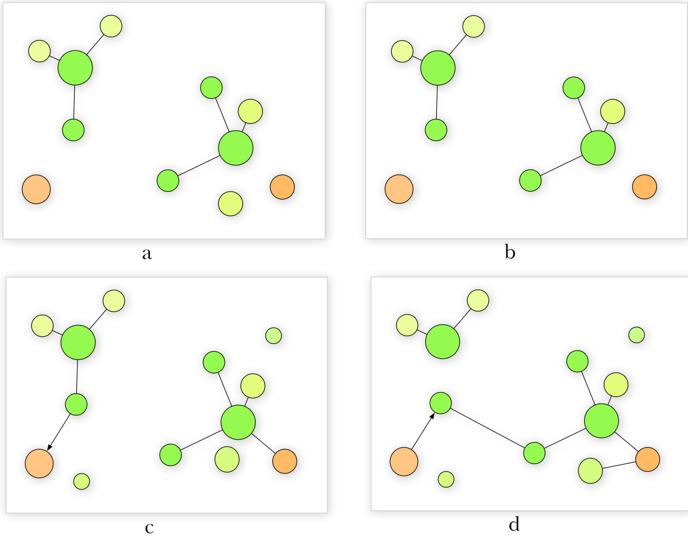

Figure 1. These panels show successive changes occurring to an initial graph (a). First, a node is removed (b); then, two new edges are added, along with a number of other nodes (c). Finally, one subgraph becomes disconnected from the rest (d).

creates difficulty for the user in anticipating when events they need to be aware of are likely to occur.

2.2. Change Blindness

Even when the user is paying attention to the view, they may not be aware of everything that is happening, due to a perceptual weakness known as change blindness [9]. When faced with a constant flow of information, a user may be too focused on one area of the view, may not recognise what they are seeing or may miss visual changes entirely. Cognitive science has addressed the issue of minor or even major changes going unnoticed by observers [13].

The writer Alan Moore has argued that one of the benefits that both novels and comic books have over films is they can be taken at their own pace rather than the pace mandated by the flow of the film [7]. In these media, the reader can slow down to fully take in a complex part of the book, speed up during passages of little consequence, and return to previously-read pages at any time to clarify facts and increase understanding.

In a visualisation observer's case, this "pace" is the rate that the data was recorded and is played back at. However, since we engage with dynamic visualisations in a different way to the passivity of watching a film, we have the opportunity to make understanding the passage of a visualisation easier, by enacting some control over the speed that the data is presented at.

3. EVENT SEQUENCES

We suggest that it would be useful to lessen the cognitive load on a user in a second way; by affording them additional time to make sense of the changes to the graph, and breaking up the constant stream of modifications into discrete blocks of changes.

When developing a dynamic visualisation, the developer will typically load in a trace of changes, or mutations, which occurred to the data structure. This trace may exist as a file with a known size, or may be read into a buffer from where it is being generated, and then polled asynchronously and fed into the visualisation. Here we'll use a file of ordered mutations, with one per line, describing the generation of a graph. The time-stamped list of mutations take the form

```
010704120856-0700 add node1

010704120858-0700 add node2

010704120867-0700 link node1 node2
```

We refer to each of these lines as *events* in the system, and the queue of events in the input file as an *event sequence*. The visualisation will render the data structure as it stands, and then a loop which continually reads the input file for new events will read in the next few events for that time step. This might be a single event, many events all happening in a short space of time, or nothing for that time step, depending on the data. The visualisation will continue until it reaches the end of the file.

There may be some other changes in layout, colour or other visual effects which come about purely through adaptation by the visualisation engine. For example, as a new node is added to the graph, nearby nodes may be moved around so that they maintain a certain distance between themselves. Throughout all of these changes, the user is at the mercy of the data: the visualisation will continue to render new events at the rate and rhythm that is defined in the input file, leaving the user to keep up with this pace or miss potentially important information.

This imposition is not ideal for a number of reasons. First of all, events are wont to occur at a volatile rhythm, which may be rapid if events are clustered together in the trace. This means the visualisation will be updated frequently, without regard for the difficulty the user is having at incorporating previous events into their mental model. To take an example from life, the reader may have themselves experienced a seminar speaker proceeding through material at too quick a pace, and the watching attendees wishing they would slow down their delivery so that they could properly note down the important concepts without fear of missing something. Conversely, there may be long passages of time when nothing of much consequence is happening, wasting the observer's time by giving them nothing to analyse.

3.1. Perceiving Time and Space

Humans have an internal sense of time passing, but it's well known to be inaccurate and susceptible to significant influence from many external factors, including visual stimulus, caffeine intake and so on [5]. Readers will identify the feeling of time passing quickly or slowly depending on their engagement with a task, whether they find it engrossing or tedious.

We can also fool ourselves quite effectively into believing that some things occur more quickly than they do in reality. User interface testing performed by Apple in the early days of desktop interfaces reported that users would consistently report that they felt using keyboard shortcuts allowed them to perform tasks faster, while observers timing their performance would report that those using the mouse exclusively would actually be faster [14]. This phenomenon was due to the additional time that the keyboard shortcut users needed to perform the often significant cognitive processing to remember which abstract key combination they required (reported as two seconds in some cases). While performing these high-level cognitive functions, the user simply did not notice time passing. Though this effect is certainly less pronounced now due to people's increased familiarity with computer interfaces, the gulf between a user's reported subjective perception and the absolute numbers from the stopwatch is striking, and shows the fragility of our internal model of time.

On the other hand, our visual perception is highly adept at accurate local comparisons. When presented with two parallel lines, observers can instantly see which is the longer of the two.

We seek to invert the balance here: in the application of our *Showtime* tool, we take time out of the equation by only allowing visual data changes to occur at certain regular, predictable time intervals. By marshalling events in this way, the user always knows that they have a certain constant amount of time to understand a particular layout before the next change arrives. It also gives the visualisation algorithm more time than it might usually receive to optimally lay out the graph for legibility in between changes. This helps it avoid visual oscillation, harsh transitions and the ensuing damage to the user's mental model.

3.2. Visualising Time

Though events now occur at a constant rate for the user, they will not have occurred regularly in the event sequence, and these relative delays are important information for the user's understanding of the structure. The user needs to be informed of how long has passed since the last event. To do this, we introduce a visual timeline which shows the length of time between each event, so that the user gets a sense of the relative temporal gaps between events.

The timeline flows down the left side of the view, as seen in Figure 2. Each segment of this timeline is sized based on the length of time it represents. Traditional timelines are presented horizontally, with notches corresponding to the values being represented. If presented vertically as in this component, the user can easily do a visual comparison on the lengths of two segments. Another reason for favouring a vertical timeline is that when we place this to the side of a widescreen monitor as it is in Showtime's layout, the screen area left over for the network visualisation has an aspect ratio closer to that of a square. The first segment at the top of the screen represents the unit length of a period in the simulation (this is currently set to two seconds), so the observer has a frame of reference for visual comparison.

Besides encoding the relative lengths of time between events, the timeline acts as a visual history of the data stream. The user can visually inspect the timeline to see if the events are happening close together or far apart, and at a regular or chaotic rhythm, but they don't have to deal with any chaos or long waits themselves.

This non-linear time series technique can be applied to any data source, whether for offline analysis of a data file, or for analysis of data being streamed directly from a data source. This technique is inappropriate for visual analytics tasks that are time-critical. Events will be buffered into a single continuous feed of changes, and will take longer in real-time to go through if there are many events bunched close together. By the same token, the viewer can watch more data then they could in real-time if the event queue is sparse. This acts in a give-and-take fashion: when events occur frequently, the visualisation expands this time and takes longer to get through these events while buffering future events. Then when

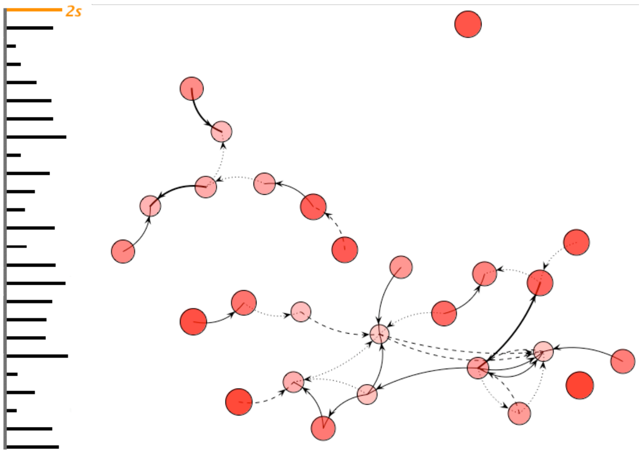

Figure 2. The timeline is presented vertically, and the latest time segment is added to the top after each event in the sequence. Old events scroll off the bottom of the screen.

nothing much is happening in the trace, it can "catch up" to more contemporary events.

Some data sources are more appropriate than others. For example, a social network analysis is unlikely to ever be analysed in real-time, as it happens over such a long time-frame. The type of patterns you're looking for are more appropriate to this longitudinal view. In some data sets, due to the resolution of the recording instruments, many events may share the same time-stamp. In these cases, we can separate the events in time so that they can be seen individually.

We investigated methods to draw a user's attention to the changes as they were introduced into the visualisation. We do this by adding a perceptual layering effect, and giving some elements more visual weight than others, which creates a separate visual foreground and visual background. We increase saturation and brightness to highlight nodes that have recently been affected by change, and gradually fade to the colour of the rest of the nodes over the course of five change cycles. This allows changes that occur close to previous changes to be recognised as potential patterns by the observer (spatial perception being the most salient perceptual channel known to be available to information visualisation designers [8]).

4. DISCUSSION

The extra time that the layout algorithm gains to move nodes around is particularly useful for iterative algorithms like force-directed algorithms. Our study of this technique so far suggests that two controls are needed: an overall context to show where in the timeline you are currently, and thus how much has progressed and how much is left; and a control to do a quick rewind over the last few cognitive steps—essentially for the same use case as when a viewer misses a line of dialogue or a subtitle in a film.

It would be useful in future if the event system had a more semantically-rich understanding of the events in the system, in a similar way to how semantic zoom systems work [10]. For example, it could respond not just to the density of events in the event sequence, but also to the provenance and relevance of these events to the system as a whole. This would elide more trivial details to focus on the key events in the sequence.

5. CONCLUSIONS

We have presented *Showtime*, a non-linear time interpolation technique that augments traditional dynamic visualisations by introducing a visual element that represents segments of the passage of time. This allows us to speed up the replay of the evolution of a data set: rather than having to watch the events occur in real time, we can skip between them while the time sequencer visually shows the speed that time is passing. This allows us to order events into a continuum where changes to the data structure (usually a graph) occur deterministically on a set rhythm, and quiet periods where no events are occurring can be quickly skipped past.

6. REFERENCES

1. J. Branke. Dynamic graph drawing. Springer Lecture Notes In Computer Science, pages 228–246, 2001.
2. P. Eades, W. Lai, K. Misue, and K. Sugiyama. Preserving the mental map of a diagram. Proceedings of Compugraphics, **91** (9):24–33, 1991.
3. M. Farrugia and A. Quigley. Visual data exploration of temporal graph data. In Conference on Visualization and Data Analysis 2009, San Jose, California, USA.
4. C. Friedrich and P. Eades. Graph Drawing in Motion. Journal of Graph Algorithms and Applications, **6**(3):353–370, 2002.
5. C. Harrison, B. Amento, S. Kuznetsov, and R. Bell. Rethinking the progress bar. In UIST '07, pp. 115–118, New York, NY, USA, 2007. ACM.
6. Y.-Y. Lee, C.-C. Lin, and H.-C. Yen. Mental map preserving graph drawing using simulated annealing. In APVis '06, pp. 179–188, Darlinghurst, Australia.
7. A. Moore. Writing for comics. Avatar Press, January 2007.
8. T. Munzner, F. Guimbretiere, and G. Robertson. Constellation: a visualization tool for linguistic queries from mindnet. Information Visualization, 1999. (Info Vis '99) Proceedings. 1999 IEEE Symposium on, pages 132–135, 154, 1999.
9. L. Nowell, E. Hetzler, and T. Tanasse. Change blindness in information visualization: A case study. In INFOVIS '01: Proceedings of the IEEE Symposium on Information Visualization 2001 (INFOVIS'01), page 15, Washington, DC, USA, 2001. IEEE Computer Society.
10. K. Perlin and D. Fox. Pad: an alternative approach to the computer interface. In SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques, pages 57–64, New York, NY, USA, 1993. ACM.
11. H. C. Purchase. Which aesthetic has the greatest effect on human understanding. In Graph Drawing, Rome, Italy, September 18-20, 1994, pp. 248–261. Springer, 1998.
12. R. Shannon, G. Williamson, A. Quigley, and P. Nixon. Visualising network communications to evaluate a data dissemination method for ubiquitous systems. In UbiComp 2007 Workshop Proceedings, pages 288–291, Innsbruck, Austria, September 2007.
13. D. Simons and C. Chabris. Gorillas in our midst. Perception, 28:1059–1074, 1999.
14. B. Tognazzini. TOG on Interface. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1992.
15. E. Tufte. The Visual Display of Quantitative Information. Graphics Press, 1983.
16. C. Ware. Information Visualization: Perception for Design. Morgan Kaufmann, 2004.