

Clustering Provenance

Linus Karsai

A thesis submitted in partial fulfillment of
the requirements for the degree of
B.I.T (Honours)

Graph Visualisation
University of Sydney



June 2016

Acknowledgements

First of all I would like to thank the three academics who I never would have completed this thesis without. To my supervisor Alan Fekete, you joked with me about saying that you contributed very little, but I would have to disagree. Over the last year I never felt unsupported or without direction; you were kind, empathetic, and above all incredibly intelligent. Your ability to consolidate and distil my typed ramblings will never cease to amaze me. To Judy Kay for your guidance and direction, particularly in all things usability; you always made time to give feedback and read over my work. Lastly to Paolo Missier my foreign correspondent, expert in provenance and who convinced me over a year ago that this would be an interesting project (you weren't wrong!)

Thank you to my parents Gaby, Peter and Andy for accepting that I now live at my university desk, but still fed me left overs when I got home and made conversation sleepy eyed and pyjama donned. Also a shout out to my awesome sister Lucy who would leave me surprise doughnuts to find on the kitchen counter.

James, thank you for hanging around at uni late to keep me company and for eating more dinners out than is technically healthy. Vivian, my hallway buddy, thank you for constantly enforcing breaks and convincing me that I should one day take up yoga.

Lastly to all my weekly distractions. Thank you for reminding me that there is other things in life outside of work. To everyone at Randwick for making me laugh. And to everyone at board games for keeping me sane. A special thanks to Reuben, Mitch J, Mitch DM and Shu for sharing the load of weekly hosting (even though you ate all the dumplings that one time I couldn't come).

Abstract

As digital objects become increasingly important in people's lives, people may need to understand the *provenance*, or lineage and history, of an important digital object, to understand how it was produced. This is particularly important for objects created from large, multi-source collections of personal data. As the metadata describing provenance, Provenance Data, is commonly represented as a labelled directed acyclic graph that can get large and unwieldy, the challenge is to create effective interfaces that allow the simplification of these graphs in order to make them more accessible to everyday users.

I approached this problem by designing two methods of manually *clustering* and implementing them in to the ProvOwl prototype. A comprehensive usability study was conducted in order to evaluate the effectiveness of these clustering techniques.

Contents

Chapter 1. Introduction	1
1.1. Overview	1
1.2. Related work	3
1.3. Contributions	18
Chapter 2. Clustering Interface Design	20
2.1. Manually clustering nodes	21
2.2. Challenges	23
Chapter 3. Working Prototype	26
3.1. Functions and Design	27
3.2. Implementation	32
Chapter 4. Usability study	38
4.1. Study Design	38
4.2. Study Results	45
Chapter 5. Conclusions and Discussion	49
5.1. Future Work	49
Bibliography	51
Appendices	53

CHAPTER 1

Introduction

1.1. Overview

Provenance as a concept has been around for a long time. It originally comes from the Latin word *provenio*, meaning “to come forth” and its primary use has been in the field of art and antiques where provenance is used to identify the authenticity and quality of a piece of artwork. For anyone with a background in databases this term may also be familiar as the concept of linking tuples from the output of a query to the reason they exist.

Digital provenance is a type of metadata representing the lineage of an object. Naively it can be compared to the information stored in revision history systems like that used in Google docs¹ or version control systems such as Git or Mercurial². These systems are usually limited to storing revisions and authorship, provenance extends beyond this by making it possible to store and ultimately trace what other entities and activities led to a digital objects current state. By studying the provenance of a digital object a user is able to identify what processes, other files and bits of data influenced the object.

Provenance has the potential to be useful in any field where the lineage of an object needs to be stored. One example would be in data warehouse systems where information about data transformations and data origins is stored. Another would be banking or accounting systems where research has been done into preventing provenance forgery through the use of block chains [16]. The field that is the focus of this paper is that of personal data management, using provenance to allow users to understand how their data has been used. Often when a user chooses to share their personal data it is aggregated, anonymised or passed through any number of functions. By exploring provenance users have the ability to understand exactly how much and in what state their personal data is used.

An example to illustrate this point. We have a person called Alice. She tracks her steps using a Fit-Bit³ and weight using a Withings scale⁴. She uses the Fit-Bit dashboard to monitor her progress, it displays a series of summaries related to her fitness data

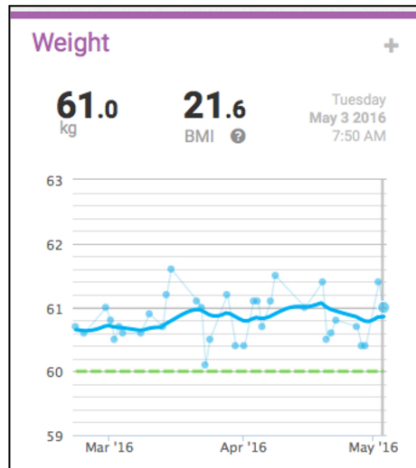
¹Google docs: <https://docs.google.com>

²Git <https://git-scm.com/> and mercurial <https://www.mercurial-scm.org/>

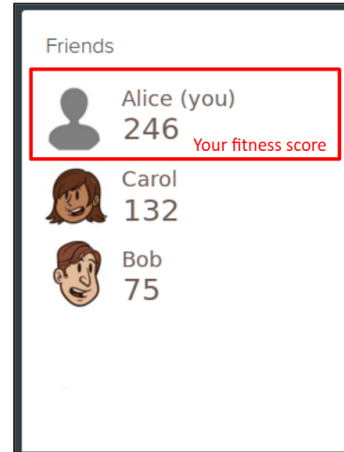
³Fitbit is a company that sells wearable devices to track you steps and other fitness data <https://www.fitbit.com/>

⁴Withings sells digital scales that can log you weigh information wirelessly and automatically <http://www.withings.com/us/en/products/smart-body-analyzer>

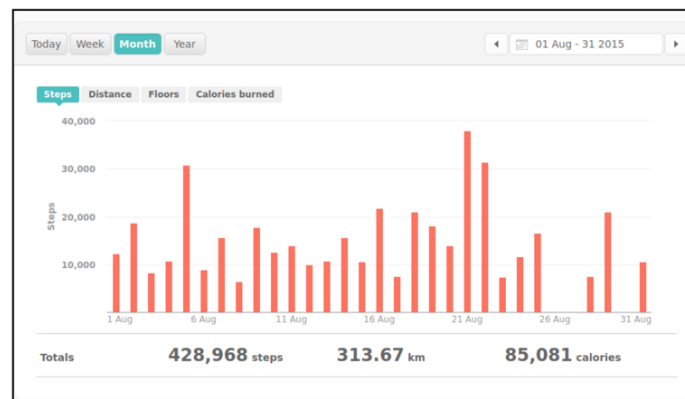
as seen in Figure 1. However she wonders how some of this data is calculated, from (B) in Figure 1 she had deduced that other people can see her fitness score and she is not comfortable with what information her friends might be able to imply from it (for example, would her friends be able to tell if she has missed her fitness goals?).



(A) A summary of Alice's weight data.



(B) A display of Alice's current fitness score and the scores of her friends Bob and Carol.



(C) A summary of Alice's step data.

FIGURE 1. Alice's Fitness dashboard shows summaries of information about her. She wonders how some of this information is calculated.

Luckily Fit-Bit stores the provenance of these summaries, so it is possible to find out what data is used to create each of those in Alice's dashboard. However the provenance is stored in PROVN, a text serialisation that is difficult to read, as seen in Figure 2. Alice, a non technical user, is not surprisingly overwhelmed by this and is no closer to understanding how her fitness score was calculated than before she had access to its provenance.

The industry standard for visualising provenance graphs is through acyclic directed graphs (although some other visualisation techniques have been experimented with [6]). Representing Alice's fitness score provenance this way makes it easier to understand

than the raw PROV-N file. As seen in diagram (A) of Figure 3. It is now possible to see that across the bottom of the graph is a series of entities attributed to Alice such as `step_tracker_data` and `step_goal`. It can also be seen that these entities are somehow used, through a series of activities, as input for her fitness score. However this graph is still too complex for a non technical user and perhaps even a technical user. Therefore the next step is to simplify the graph by clustering some of the nodes together to create something like diagram (B) of Figure 3. In this graph I have taken all the nodes related to weight data and step data and created clusters of them both, reducing the graph from having 15 nodes to only five. This makes it much easier to understand and conveys the same core concepts as before except through fewer nodes. We come here to the crux of the problem that my thesis explores: *Provenance graphs are too complex for regular users to understand.*

What we describe in this paper is the design, implementation and evaluation of an interface that enables users to cluster effectively, and in turn simplify graphs in order to make provenance more approachable.

1.2. Related work

While the provenance field is quite new there is still a substantial body of work researching it. Below I outline the five main categories of provenance research: overview, acquisition, storage, security and display. I then describe what contributions my work has to the field.

1.2.1. Provenance Overview. As mentioned above, provenance is historically used in the field of antiques as a guide to authenticity of quality. In this thesis we focus on the more modern concept of digital provenance, a metadata that stores the lineage of a file. Only recently defined, below are two papers outlining the requirements and definition of digital provenance.

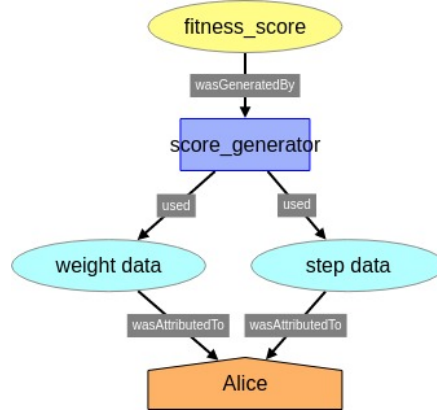
“A Primer on Provenance” Carata et al. [10]

```
1 agent(ir:Alice, [fullname="Alice Andrews"])
2
3 entity(ir:fitness_score)
4 entity(ir:weight_success)
5 entity(ir:aggregated_weight_data)
6 entity(ir:weight_goal)
7 entity(ir:weight_scale_data)
8 entity(ir:weight_manual_data)
```

FIGURE 2. A extract of the provenance of Alice’s fitness score written in the provn standard. It is quite unwieldy to read, even for someone similar with provenance data. View the entire file in Appendix D



(A) The provenance of Alice's fitness score represented as an acyclic directed graph.



(B) The provenance of Alice's fitness score with manually created clusters.

FIGURE 3. Two directed acyclic graphs showing the provenance of Alice's fitness score.

Published in the *Communications of the ACM* magazine, this article gives an in depth introduction to digital provenance. It presents current research in the field from a practical perspective and talks at length about their possible use cases as well as issues obtaining, using and securing provenance.

This paper does an impressive job of laying out the fundamentals of digital provenance, as well as identifying current research efforts and areas of improvement. It also describes the language and terminology used by the provenance community when discussing different features of provenance, such as the concepts of *granularity* and *layering* when collecting provenance.

“Requirements for Provenance on the Web” Groth et al. [14]

This paper from *The International Journal of Digital Curation* synthesizes three main categories that can be used to assess the adequacy of different online provenance systems. It splits these categories into multiple dimensions for finer grained analysis.

The three main categories this paper uses as a base for assessing provenance systems is: content (the data itself and information stored in it), management (different ways of collecting and storing the data) and use (different ways of using the data, visualisation etc.). Each of these categories is then divided into a multitude of related dimensions. For example the *Content* category includes versioning (records of changes written to artefacts over time) and attribution (sources that contributed to making an artefact) as dimensions. The papers support their claims for the requirements of these dimensions by describing them in relation to three scenarios. The scenarios are well picked in order to mirror real world applications.

This paper outlines a range of dimensions that can be used in order to assess an interface. In terms of HCI these can be used as the base for tasks to test on users as well as suggesting features that may be useful in my application. For example, one of the dimensions for *Use* is *Imperfections*, suggesting that I need to take into account that erroneous/imperfect provenance data may exist when creating my interface.

1.2.2. Provenance Acquisition. Provenance can be acquired in a variety of ways. A common goal in the literature is to collect provenance automatically and transparently in order to reduce cognitive load on users as well as reducing erroneous or false data. The papers below discuss the collection of provenance across two different layers: operating system level and application level, both with their advantages and disadvantages.

1.2.2.1. *OS level provenance acquisition.* OS level provenance acquisition stores the calls between different processes and files. It has the advantage of clearly showing what applications rely on which resources and can be used via a custom kernel to log provenance on a computer without requiring any changes to the users applications. However because of its intrinsic need to treat processes as “black boxes” it can also trigger false positives when graphing lineage. For example a process may invoke the use of multiple libraries however it is arguable as to whether they’re all relevant in terms of provenance.

“Provenance-aware Storage Systems” Muniswamy-Reddy et al. [20]

Published in 2006, the PASS paper is often cited in related literature as a prime example of OS level provenance collection. This paper discusses the advantages of having provenance information maintained by the storage system as well as presenting a PASS implementation with analysis of its performance costs.

In most implementations provenance is stored in a standalone database systems or flat files. This paper argues that provenance should be maintained by the storage system since it is the storage system that manages existing metadata for files. This allows tighter coupling between data and provenance as well as allowing the transparent collection and management of provenance information. The paper lists a set of requirements for a PASS system as well as their own implementation. They benchmarked their implementation on a 500Mhz Pentium 3 computer with 768MB of RAM. The two primary benchmarks used are small and large file microbenchmarks. In the case of the small file benchmark the overhead associated with creating and writing files was as much as 200%, however it is noted that the absolute numbers are still quite small and that this benchmark in particular is quite challenging to PASS. Results from the large file benchmark were better with a time overhead of no more than %10-%20. It seems more reasonable that real users will only encounter overheads of the latter in daily use. Overall the PASS system provides extra functionality not currently available in other systems with only a moderate overhead. It is also important to note that the systems used for these benchmarks are a lot less powerful than what average users use today.

1.2.2.2. *Application level provenance acquisition.* Application level provenance acquisition has the advantage of creating more relevant relationships between different entities. However it suffers from the overhead required to implement effective capturing and in a lot of cases onus on the user to annotate their actions. Most of the papers below discuss a system that requires developers to somehow modify (usually marginally) their application in order to correctly collect provenance.

“BURRITO : Wrapping Your Lab Notebook in Computational Infrastructure” Guo and Seltzer [15]

Authored by Margo Seltzer (one of the top contributors of the field[18, 24, 6, 19, 7],) and Philip J. Guo, this paper presents one of the many ways to collect provenance on a computer. It focuses primarily on research related provenance and its application in respect to *lab notebooks*.

The application presented in this paper, Burrito, consists of two parts: an extensible platform that automatically captures provenance as well as a set of applications that allow annotations and querying of that data. Stored in a local MongoDB, provenance is captured through GUI window interactions, OS-level capture (like PASS[20]) and integration with a versioning filesystem. On top of this is a series of plugins that allow provenance to be captured by the following applications: microphone, xpad, Firebox, Chrome, Vim, Bash, Python and the clipboard. Burrito then provides four applications that allow interaction with the provenance data. An activity feed that sits on the users desktop, presenting provenance events chronologically and allowing annotation of events as they happen. A computation context viewer that allows exploration of a files changes and how they impacted output. An activity context viewer that shows what was been read and written at the time of an activity. Finally they also provide a Lab Notebook generator that creates a HTML file summarising the users activities.

Over a month this system accumulates approximately 2GB of data and although no benchmarks are reported it is stated to have minimal overhead.

This paper is interesting as it brings provenance to the forefront of the users mind. The goal been that they are constantly annotating and referencing past provenance activities during the length of a project. It also has the added advantage of having images and other program output inline the visualisations.

“A general-purpose provenance library” Macko and Seltzer [18]

Written in 2012 the authors of this paper Macko and Seltzer are both heavy contributors of the provenance field. Whilst most of the other provenance frameworks mentioned in this review are limited to a particular workflow or language, Macko and Seltzer argue that the real world is a lot more heterogeneous and to accommodate for this they argue the need for a general-purpose provenance library.

This paper presents the Core Provenance Library (CPL) a C++ application that allows the integration of provenance storage into any application. It supports bindings for C, Java and perl as well as a command a line tool for creating shell scripts. The provenance is stored on a database system of your choice through two drivers; these where tested to work for MySQL, PostgreSQL and 4store. The main disadvantage of this framework is that it is the onus of the developers to implement calls to the provenance framework in their application. Multiple features are implemented in order to allow easier integrating such as automatically taking care of persistent storage, cycle detection and resolution, as well as supporting query visualisation through map orbiter [24]. However developers/project managers need to take into consideration integration when developing an application.

This library provides hope for a standardised way of capturing provenance information. However as mentioned above it is up to developers to correctly integrate it and there’s an intrinsic trust that correct and truthful provenance will be recorded. Disappointingly the google code repository hasn’t been updated since 2012 and no replacement github repository has been created in light of the shutdown of Google code, suggesting that the project is no longer maintained.

“Capturing Provenance in the Wild” Allen et al. [2]

This paper describes the ability to compose multiple provenance-unaware services in an “open world” system and collect provenance information about their execution. It presents a provenance collection application that sits upon the enterprise service bus (ESB) in order to log provenance information from communicating applications.

This paper advocates for a provenance collection solution that doesn’t require system-invasive strategies such as custom kernels or modified applications. It argues that in order to be useful in the “open world” provenance capturing tools must fulfil the following tasks, capturing provenance:

- across multiple systems with no assumption of control.
- from legacy systems
- at the level of application interaction, *not* foundational technology stack.

In order to evaluate their ESB MULE Capturing Agent (MCA) they tested it upon two different workloads: Loan Borker (a standard MULE test scenario) and CoTLooper (a scenario that uses cursor on target messages). The average time to log provenance within MULE over time decreased as the more provenance nodes were created eventually evening out around $1 \times 10^8 ns$. However if message reflection (a method for examining runtime behaviour of applications in the Java VM) was required to gather data from the payload of the message then message capture time can be greatly increased. Whilst these metrics are useful in identifying the overhead of using an ESB to capture provenance information from communicating applications (in this case minimal), it is arguable that the provenance information would suffer from problems similar to OS level capturing systems, creating false positive edges in the graph.

This paper is highly cited in the provenance community. It shows one of the many *layers* of provenance that can be recorded and is one of the few systems that records provenance across multiple machines.

1.2.3. Provenance Storage. Once provenance has been captured by one of the methods mentioned above, it has to be stored somehow. In a lot of early papers the provenance was stored in different proprietary flat file formats. Since then a standard has been created by the W3C, as well as research into how provenance can be stored on the cloud.

“PROV Model Primer” Belhajjame et al. [4]

This online document is an initiative by W3C to create a standard as by which to discuss and store digital provenance. This paper is part of a series of documents outlining different aspects of the PROV standard, in particular it is a primer on the fundamental PROV concepts.

This document outlines the three main concepts illustrated in Figure 3, as well as different serialisations in languages such as XML, JSON and PROV-N. Each of the concepts are briefly described as:

- Entities: Physical, digital, conceptual objects
- Activities: Elements that cause an entity to come into existence
- Agents: Someone or something that can be assigned responsibility for an activity taking place.

I used the standards defined in this paper as the oracle definition for provenance in my designs. My prototype was built so that it can read provenance files in the PROV-N format.

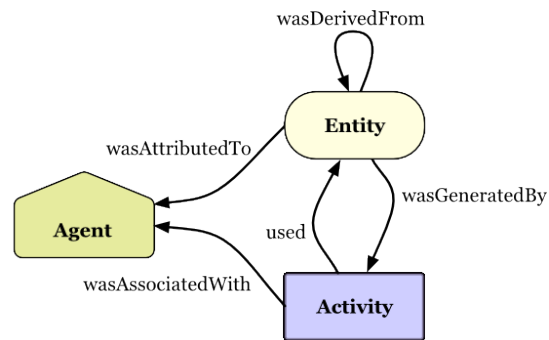


FIGURE 4. Key concepts and relationships from the PROV standard displayed in a labelled acyclic graph.

“Provenance for the Cloud” Muniswamy-Reddy, Macko, and Seltzer [19]

This paper presents the following problem: although cloud data and provenance both exist, they are not usually used in conjunction because it is difficult to store both in the same location. Traditionally if data is stored in a database, the related provenance information is then stored somewhere else. To mitigate this they present 3 different methods for maintaining data and provenance in current cloud stores.

This paper’s contribution is three protocols (using off the shelf software,) each satisfying a different number of properties crucial for provenance systems: provenance data coupling, multi-object casual ordering, data-independent persistence and efficient querying. Each of the systems work with minimal overhead and surprisingly the protocol that fulfils all four of the afore mentioned properties runs as well if not better than the other protocols. The benchmarks used on each of the protocols where varying in their workloads, ranging from CSV backup simulations to workloads representing scientific computation. There is a diverse selection of workloads that simulate scenarios that provenance would most likely be used in.

This creates an avenue of further research. Currently my prototype interface only reads locally stored flat provenance files. In the future it would be useful to expand this to provenance stored on the cloud as well.

1.2.4. Provenance Security. Once provenance is captured and stored the issue of securing it is paramount. Simple per-attribute access permissions are ineffective for provenance entities because so much information can be gathered from relationships. It is also worth considering how to secure the data in such a way that history can’t be “re-written”. The below papers discuss this problem in greater detail.

“Securing Provenance” Braun, Shinnar, and Seltzer [7]

A discussion paper from 2008, Margo, Uri and Avraham focus on the issue of properly securing the access rights of provenance data. It frames the problem and identifies issues requiring further research.

This paper aims to start discussion and highlight the importance of considering provenance security during the implementation and development of an application. It proves that provenance cannot be secured in the same way as traditional data because of its ability to provide information through relations as well as the artefacts themselves, arguing instead that provenance needs its own security model. The paper suggests that there are three main aspects of provenance that need to be secured, each in their own way: the *node data*, the *relationships* between nodes and the *attributes* related to nodes or relationships. It gives concrete examples of why each of these aspects need to be protected and why regular access control methods aren't satisfactory.

This piece is though provoking in the issues it outlines related to securing provenance. It raises the question of how to present partial graphs caused by users with limited permissions. A problem also worth considering in future work is that of how to present to a moderator which bits of a graph are accessible by who.

“A formal framework for provenance security” Cheney [12]

A common theme among papers regarding provenance systems is that they agree that provenance is an open and unsolved issue. This paper from the University of Edinburgh outlines a formal model for provenance, formalization of security fundamentals *disclosure* and *obfuscation* and exploration of its implications in various domains.

This paper identifies five main classes of provenance policies: availability, confidentiality, integrity, reverse-engineering and explanation. They go on to describe a high level security framework focusing on the availability (ensuring information about input is available to users) and confidentiality (ensuring that confidential information is never disclosed to a user) of provenance data. This framework is general enough to not be specific to any one system, meaning its rules and observations can be applied to any system. The paper provides support to its claims through detailed mathematical proofs. It also provides three instances of how the provenance security framework can be used to create provenance policies.

Unfortunately this paper provides limited benefit to my project because its focus on security is abstract enough to be most useful to researchers designing provenance security systems. However it provides understanding of the security issues involved as well as ideas of how security will be implemented.

“Preventing history forgery with secure provenance” Hasan, Sion, and Winslett [16]

This paper is often referenced because of its work in creating *Sprov*, a secure provenance storage method.

The author's main contribution from this paper is *Sprov*, a prototype of the secure provenance primitives, implemented as wrapper functions for the standard I/O library in C. They use a chain encryption method in order to accomplish their primary security goal: stopping undetected rewrites of history. In order to evaluate the *Sprov* library it was run through a number of benchmarks that simulate different deployment settings.

They tested two different configurations for chain storage, both recording straight to disk and also storing on RAM with a chrom daemon periodically flushing the chain to disk. Postmart benchmark was used with a dataset containing 20,000 files ranging in size from 8KB to 64KB. Different write loads were tested all the way from 0-100%. In its worse case scenario of 100% writes, Sprov had an overhead of 25% using disk storage and 11% using RAM. Small and large file microbenchmarks were used to measure the overhead as a percentage of file size. Interestingly smaller files had a larger overhead, this is thought to be attributed to disk caching (similar results have been found by other provenance benchmarks).

This paper shows one of the many ways provenance data can be used and recorded. Its relevancy to my project is in understanding how provenance will be stored and protected as well as issues that arise in a user interface only showing partial data.

1.2.5. Provenance Display. Lastly we come to the field of thesis. How to display provenance. As mentioned previously, provenance never reduces in size, it is forever expanding and been added to. Issues quickly arise when trying to present such vast amounts of information. The primary issue been that users can't process such large graphs effectively.

1.2.5.1. *Visualisation fundamentals.* There's a series of fundamentals that have been tried and tested in the field of visualisation as important rules to follow when developing visualisation tools. Ben Shneiderman discusses fundamental tasks that should be accomplished by visualisation exploration software in order to be effective. A HCI paper then presents how non-technical users view provenance information.

"The eyes have it: A task by data type taxonomy for information visualizations" Shneiderman [25]

Published in 1996 this paper is seen as one of the core papers cited when discussing complex visualisation software. It recommends seven properties that should be implemented by an advanced graphical user interface: overview, zoom, filter, details-on-demand, relate, history and extract. It distils these tasks into what's known as the Visualisation Information Seeking Mantra:

Overview first, zoom and filter, then details-on-demand
Overview first, zoom and filter, then details-on-demand
Overview first, zoom and filter, then details-on-demand
etc.

Each of these tasks is explained in great detail with arguments as to why they are necessary. It also provides examples of how different applications have implemented certain tasks, providing an array of examples to research in regard to my project.

I used this paper as a primary starting point when creating my prototype. It was effective as a means of outlining the scope of my application, defining what features where and where not important.

“Provenance for the People : An HCI Perspective on the W3C PROV Standard through an Online Game” Bachour et al. [3]

A Human Centred Technology research project, this paper focuses on collecting feedback from non-expert users on their understanding of provenance. Through the medium of an online game they explore how well users understand the W3 prov standard[4] and its related artefacts as well as their general opinions and beliefs on provenance data.

The main contributions taken from this paper involved the feedback users gave on the representation of provenance information: one example I found particularly interesting is that a lot of users found the directional arrows confusing. The PROV standard is historical, so directional arrows between nodes are used to represent that something came from something else, however most users found this counter-intuitive and believed that the arrows should have been the other way around to show chronological ordering. The researchers use a game they created based in an Orwellian future in order to explain the concept of provenance. Users were presented with provenance graphs that they had to identify mistakes in. Eight players agreed to hour-long phone interviews and 41 submitted online questionnaires. Although the use of an online game as a usability study is unconventional its merits are well argued in the paper as well as citing other research papers that have used similar methods.

This paper provides some useful feedback from non-technical users as to what issues they perceive when dealing with provenance graphs and allows me to start my interface designs with preliminary user feedback. In conjunction to heuristic feedback the paper also provides some insight into non-expert users opinions on the security/privacy issues of provenance as well as their understanding of its real world applications.

1.2.5.2. *General Large Graph Visualisation.* Effective large graph visualisation is an on open issue in many fields outside of provenance. Both of the visualisation techniques presented below use a clustering mechanism for grouping relevant nodes.

“ASK-GraphView: A large scale graph visualization system” Abello, Van Ham, and Krishnan [1]

This paper presents an interface for exploring large-scale graphs. The visualisation tool is generic in so far as its made for the exploration of graphs in general. It uses a clustered graph to represent information with the ability to arbitrarily expand and collapse clusters to show sub-graphs.

The paper focuses on creating an interface that can present large-scale graphs to users without been impeded by system performance. This paper goes into quite some depth relating to the algorithms used to cluster nodes as well as methods used to maintain

usability with constrained resources (although it is important to note that this was written in 2006 and the issue of constrained resources may not be as relevant today). It also explores issues related to naming clusters: the authors decide on a tag based labelling system that they state is sub-optimal for understanding the contents of a cluster. No formal usability study was undertaken, the authors note that the application was in continuous use by data analysts over the six months preceding publication. It is promising that preliminary feedback was collected in this time (such as users wanting to annotate sections of the graph) and implemented into the application.

This paper's main relation to my research includes the problems tackled when visualising such large graphs. Design decisions related to clustering and pre-processing provided to be useful in creating my interface as well as issues presented in labelling clusters.

“Navigating hierarchically clustered networks through fisheye and full-zoom methods” Schaffer et al. [23]

With this paper been nearly twenty years old it is obvious how much younger the provenance research field is in comparison to visualisation. This paper talks about using a novel fish-eye interface in order to show large graphs while allowing users to zoom into particular nodes and not lose context. They also provide usability study results in order to support their claim that fisheye views improve performance compared to traditional variable-zoom interfaces.

This paper sets out to research the problems created by viewing large-scale graphs in a variable zoom interface. The authors create a usability test in order to compare the time taken to complete certain tasks on two interfaces. The first is a variable-zoom interface that allows the user to control the zoom level of the entire graph and pan around to find information. The other is a fisheye interface that clusters nodes and allows expansion of clusters in order to view greater detail. The subjects were asked to act as telephone technicians and had to navigate a network graph in order to identify a broken telephone line as well as rerouting lines in order to ‘repair’ the network. The main metric for identifying interface effectiveness was time (in seconds) taken to complete tasks. Using the fisheye interface users on average completed the task 60 seconds faster than when using the variable-zoom interface. Although not statistically analysed the authors also noted that nearly one third of fish-eye results showed a near optimal route when rerouting, whilst variable-zoom results didn't come close to being optimal.

This paper raises some interesting points about the importance of allowing users to have ‘peripheral vision’ when browsing large complex graphs. There is useful details about how they chose to scale magnified clusters as well as discussion about how to accommodate overlapping clusters. Interestingly they run into similar issues as [1] when trying to label clusters, except 10 years earlier.

1.2.5.3. *Provenance display tools.* There is a diverse range of existing provenance display tools that aim to present provenance to users in an easy to understand way.

Whilst most of the applications use a directed acyclic graph to present provenance information, there is also representation through the use of Sankey diagrams and orbital graphs, which depending on what information you wish to portray can sometimes be more effective than DAGs.

“Provenance Map Orbiter: Interactive Exploration of Large Provenance Graphs” Seltzer and Macko [24]

As provenance data becomes more common, the amount of provenance information continues to expand. It is not uncommon for the size of provenance data to greatly outweigh its target file, creating provenance data that has upwards of thousands to millions of nodes. This paper presents a Java application that can be used to visualise large scale provenance graphs through the use of zooming and summation techniques.

This paper’s core contribution is a cross-platform Java application, titled “Map Orbiter”, that can be used to visualise RDF/N3 and OPM formatted provenance data. It simplifies complex graphs by grouping related nodes into *summary nodes* and allows users to view more information about a summary node by *zooming in*. It also supports the filter task, allowing users to pick a subset of nodes that they would like to view.

This application is often referenced in other provenance visualisation papers as the primary example of directed acyclic graph visualisation. It focuses on similar issues to my project, but focuses on a Java implementation and reads provenance in the RDF/N3 and OPM format.

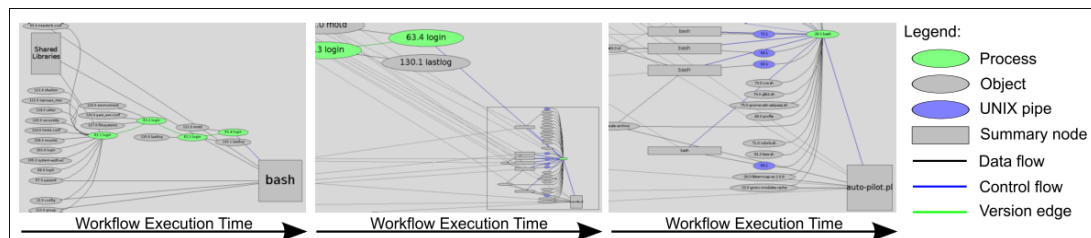


FIGURE 5. A screenshot of Map Orbiter and its semantic zoom. Left: a high-level view of Linux’s login process and its dependencies. Middle & Right: the progress of zooming into the bash summary node (in the lower-right corner), representing the login shell and the corresponding user session.

*“Zoom*UserViews: Querying Relevant Provenance in Workflow Systems” Biton, Boulakia, and Davidson [5]*

This paper is a demonstration that shows how user views can be used to reduce the amount of information returned by provenance queries. It allows users to select which parts of a provenance graph (referred to in this paper as a *work flow*) they find relevant and have the original graph simplified to only include the parts they picked, as seen in Figure 6

Similar to clustering, the ZOOM interface simplifies parts of a provenance graph by collapsing nodes, however instead of creating a cluster object, ZOOM creates a replacement *composite* node. The paper illustrates how it generates provenance information from logs stored on an Oracle warehouse as well as outlining some of the optimisations that can be used in Oracle 10.2 to aid in the generation of immediate provenance. The authors also discuss what properties they believe important when collapsing nodes into composites, what features make a “good” user view. These features include preserving the inputs and outputs from relevant nodes and hiding as much detail about irrelevant nodes as possible. In terms of size the examples in this paper are limited to graphs of 19 nodes, although extensions for larger graphs are foreseeable through extensions of the application.

This interface is similar to what I ended up designing and implementing. The primary differences are that my application uses provenance files from the PROV-N standard and that my application is designed for provenance exploration, whilst the ZOOM*UserViews system is aimed at simplifying graphs for use by other people.

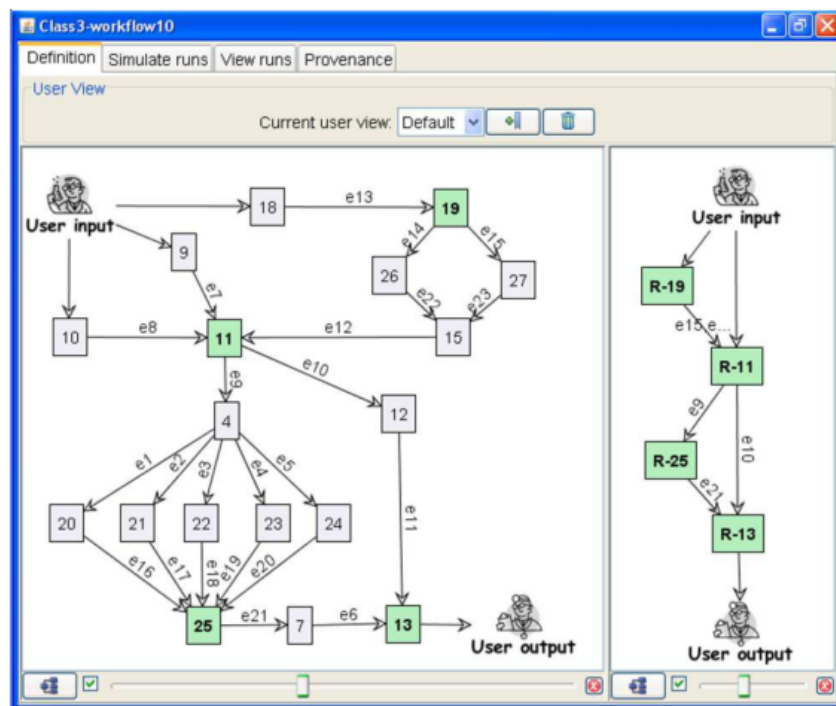


FIGURE 6. A Screenshot of the Zoom*UserViews application. On the left is the entire provenance graph with parts selected and on the right is the simplified graph created from the selected parts.

“PROV-O-Viz - Understanding the Role of Activities in Provenance” Hoekstra and Groth [17]

This paper presents an application called PRO-O-Vis that uses Sankey Diagrams in order to reflect the flow of information through activities. It discusses the application,

reasoning behind representation through Sankey diagrams and a brief evaluation of their application.

This paper's main contribution is an online tool that can be used to view provenance information stored in the PROV-O format. Once provenance has been uploaded it generates a viewer that allows exploration of the provenance by selecting an activity to focus on (from a dropdown menu) and presenting its related Sankey diagram. The interface is entirely web based and can be embedded in other websites completely self contained (doesn't require API calls to a server). A brief evaluation of the application has been made using four different sources and issues were found regarding input that had a large number of related activities, this could sometimes cause a delay in generating multiple Sankey diagrams. The paper briefly argues the use of Sankey diagrams over DAG graphs as they more easily allow visualisation of magnitude of flow within a network, comparatively complex DAG graphs can make it hard to view this information.

This application differs from other visualisation tools in the field as it's implemented completely in web tools whilst many of the visualizers run on Java. Because of this it is one of the easiest visualisation tools to install and get running because you merely have to use a web browser. I used this as inspiration for implementing my prototype as a web application.

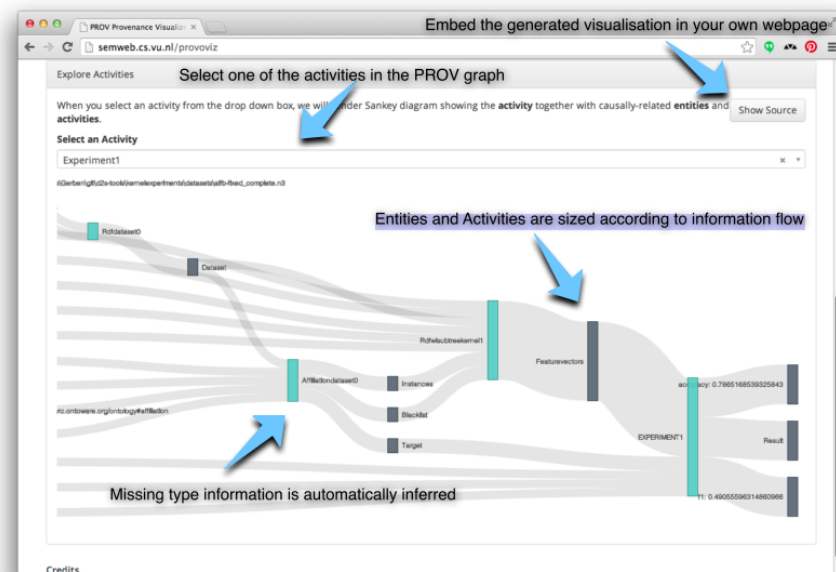


FIGURE 7. Overview report of a runtime experiment, generated by the Prov-O-Vis system.

“VisTrails : Visualization meets Data Management” Callahan et al. [9]

This is one of the earlier papers mentioning the concept of provenance as we know it today. Published at SIGMOD in 2006, this paper discusses the importance of provenance in generating visualizations. It presents an application called VisTrails that can

be used to create visualizations whilst recording provenance during exploration to aid in later scrutiny and replication.

This paper states that existing visualization tools are limited because they do not record the provenance of their outputs. VisTrails improves on the downfalls of existing visualization tools through the use of provenance. This paper was written before any provenance standards [18] so the application instead stores all relevant provenance information in an XML formatted file. The application uses provenance in order to record *evolving dataflows*, (where a dataflow is a form of visualization pipeline) allowing the modification of existing dataflows in order to compare and contrast resulting visualisations. Because provenance is captured related to changes to dataflows, it's possible at any point to check the history of a particular dataflow and identify what modifications were made to get to its current state. The VisTrails system was demonstrated through 3 examples although no references are made to any current real life uses of the application.

This paper is interesting as it's one of the earliest papers I've found in my research that identifies the importance of provenance. Its example of using provenance for visualization replication also shows how adaptable provenance can be to different scenarios, particularly its usefulness in exploratory situations. Its an important paper in order to be able to understand how the concept of provenance evolved into its current form.

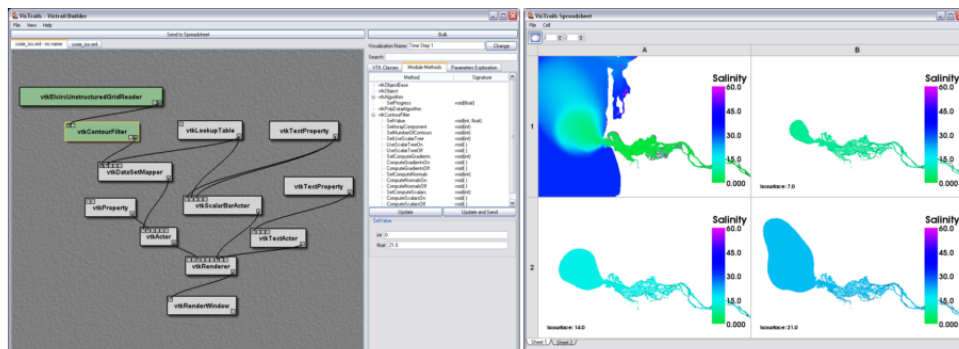


FIGURE 8. A screenshot of the VisTrails system. On the left you can see the provenance of a figure from a report. On the right is different versions of that figure that have been created.

“Evaluation of filesystem provenance visualization tools” Borkin et al. [6]

This paper focuses on the visualisation of provenance information relating to filesystem: the read and writes between processes and files. The authors implement a visualization tool called InProv and both qualitatively and quantitatively compare it's effectiveness to one of the standard provenance visualisation tools Orbiter[24].

This papers main contributions are: Firstly, qualitative research with domain experts in order to create a set of requirements for a provenance visualiser. Secondly, InProv a provenance visualization tool that uses a radial layout and chronological clustering. Thirdly, a quantitative user study comparing the effectiveness of the two interfaces

InProv and Orbiter. The qualitative research was undertaken via semi-structured one hour long interviews. They asked the domain experts to talk about how they used provenance information as well as demonstrating their daily workflows. Using results from the qualitative research they created a set of core tasks they wished to accomplish and used radial visualisation to create the InProv tool. The user study had 27 participants who were asked to accomplish tasks of varying difficulty on both the InProv and Orbiter interfaces. The metrics used to analyse performance were seconds required to complete task as well as percentage of correctly completed tasks. In most cases the difference between time to complete tasks was not significantly different between the two interfaces. However using a new clustering algorithm that clusters nodes based on time of activity (compared to process tree clustering) was shown to have significant improvement on task completion time by as much as 90 seconds.

This paper provides a lot of useful quantitative and qualitative information through its well laid out user studies. Of most interest is the interviews with domain experts. Also the choice of using *time based hierarchical grouping* is of particular interest because of the large impact it had upon improving usability. This paper makes the largest attempt at conducting a usability study and my tasks were influenced by the tasks the use in this paper.

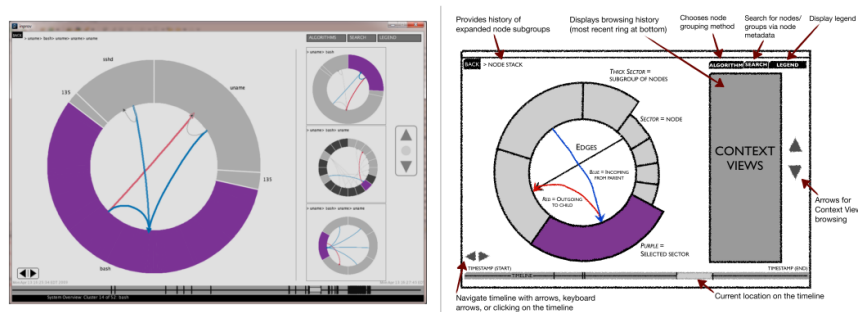


FIGURE 9. A screenshot and mock-up of the InProv visualisation interface using radial diagrams.

1.3. Contributions

The related work above shows that a growing amount of research is being conducted in the field of provenance. There's already ways to capture provenance (Section 1.2.2) as well as ways to store it once it's captured (Section 1.2.3). Research has also been conducted into how to then secure this information so that access can be controlled and to ensure provenance can not be altered arbitrarily (Section 1.2.4). However there is a gap in the research when it comes to creating interfaces that allow the simplification of graphs.

As mentioned in Section 1.2.5, there are already some existing applications that can be used to visualise provenance graphs. However most of them lack clustering and in particular clustering from a usability perspective. This leaves the following problem unsolved: *Provenance graphs are too complex for regular users to understand.*

In the upcoming sections of my thesis I outline and go into detail concerning my three main contributions to the field:

- The design of an interface that enables users to cluster effectively.
- The creation of a working prototype that implements the aforementioned design.
- A usability study evaluating the design and prototype: covering learnability, ease of use, efficiency, recovery from errors and user attitude.

Clustering Interface Design

Clustering in this paper is the action of taking a set of nodes, removing them from the graph and replacing them with a single composite node. An example of this is in Figure 1 where you can see the node *Fitness-Summary* and all its children have been clustered to create the composite node named ‘Fitness Data’. In my research I identified two primary ways of creating clusters, the first is automatically based on properties of the nodes or other heuristics, the second is via manual selection by the user. Previous research has been done into automatic clustering [6, 24], so instead what we focus on in this thesis is expanding on the work in manual clustering [5] by designing a usable interface that allows effective clustering not just as a method of graph simplification, but as a way of conveying information.

The primary requirements for manual clustering where that it be simple to use and intuitive to new users. It should be easy for users to cluster nodes together without been shown how to. But it must also be powerful for expert users, so that someone who deals with provenance on a daily basis can quickly and easily create summary graphs with composite nodes in a minimal amount of time and effort.

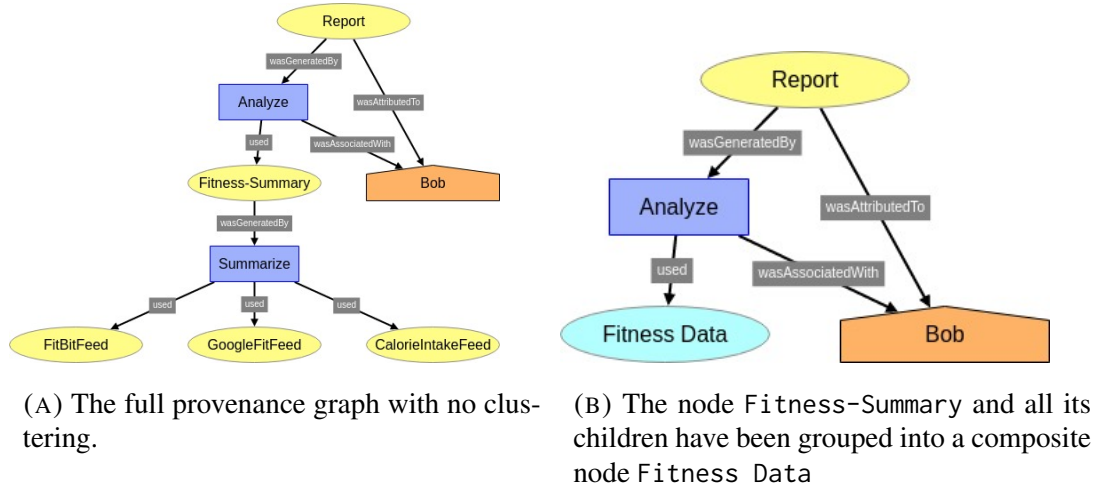


FIGURE 1. Above is a provenance graph showing the lineage of a report on Bob’s fitness. On the left is the full graph, on the right a cluster has been manually created.

To fulfil these requirements I designed two different mechanisms for clustering, one for novice users and one for expert users. Although both can be used by either category of user it is easiest to differentiate them this way.

2.1. Manually clustering nodes

The first method I designed was for novice users and works as follows: a user selects multiple nodes by clicking on them while holding down *ctrl*, each selected node is outlined red as an indicator to the user that they are selected. Users then cluster the nodes together by selecting a group function from a list of contextual actions.

This allows users to cluster nodes together with little to no prompting from an expert. However it has the drawbacks of only been viable to a small number of nodes as selecting more than a dozen or so nodes can be tedious and error prone.

In order to allow precise and powerful selection of a large number of nodes I designed a second way of allowing users to cluster nodes via a regex search function. Using the search bar users can input regex strings and matching nodes will be selected (outlined red). Once the user had tweaked the query to their liking they can group the nodes using the same technique as before; selecting the group function from a list of contextual actions. To make it easier to create multiple clusters quickly, pressing *enter* while in the search field will cluster together the currently selected nodes, this allows users to quickly create a series of clusters without having to move their hands from the keyboard (a detailed analysis of the effectiveness of this can be found below in Section 2.1.1). The regex query currently searches all properties and attributes of nodes, a future path of work would be to expand it so that you can match on specific properties.

Creating these two methods of clustering allows users to easily be introduced to the concept of clustering in an intuitive way, by clicking on nodes, then expand to the more complex and powerful tool of regex searching as their needs expand. Details on how these two methods were implemented can be found in Section 3.1.3.

2.1.1. Comparing clustering methods. GOMS is a method of predicting how long a task will take. It is outlined in “The psychology of human-computer interaction” by Card, Newell, and Moran [11]. It can be used to compare how long it takes to accomplish a task in different ways. In this case I used it to compare the speed difference between *ctrl+click* clustering and search clustering and to determine the potential benefit of one clustering method over another.

A task is given a series of letters to represent what happens when a user tries to accomplish that task. For example, using the *ctrl+click* method to select the nodes as shown in Figure 1 would be: HMPCR HK MPCR MPCR MPCR MPCR MPCR. Each letter represents something that occurs:

- K - keypress
- P - point with mouse
- C - click with mouse
- H - home hands on new device
- M - mentally prepare

- R(t) - system response time

So using this we can break down the GOMS string for *ctrl+click* as meaning the following:

- HMPCR - User moves hand to mouse and points and clicks at Fitness-Summary node (node is highlighted with red outline)
- HK - User moves hand to keyboard and holds down *ctrl* button
- MPCR - User moves mouse and clicks the *Summarise* node.
- MPCR - User moves mouse and clicks the *FitBitFeed* node.
- MPCR - User moves mouse and clicks the *GoogleFitFeed* node.
- MPCR - User moves mouse and clicks the *CalorieIntakeFeed* node.
- RMKPCR - User sees group link in contextual actions, stops holding down *ctrl*, moves mouse to the group link and clicks it. The nodes move together and create a composite node.

Similarly, using search clustering to cluster the same nodes would be represented as the following:

- HMC - User moves hand to mouse and selects the search button from the toolbar
- RMPC - The search panel is show. The user moves the pointer and clicks the text input field of the search panel.
- HMKKKKKKKKKKKKKKKKKKK - User moves hands to the keyboard and types the following regex “SummzarizeFeed”
- KR - User presses *enter*, the nodes move together and create a composite node.

We can then assign time values to each of these tasks as seen in Table 1. The times used here are based on those published in the “The psychology of human-computer interaction” book [11]. Note that these times assume an expert (familiar with the domain and application), error free user.

TABLE 1. A list of GOMS tasks and their associated timing.

Letter	Desc.	Time (seconds)
K	keypress	.08 - 1.20
P	point with mouse	.8 - 1.5 (Fitt’s Law)
C	click with mouse	.2
H	home hands on new device	.4
M	mentally prepare	1.35
R(t)	system response time	0

TABLE 2. The resulting times from GOMS analysis of the two methods for clustering a set of nodes, *ctrl+clicking* and search clustering. I assume that it takes about 1.2 seconds for the user to point with the mouse, and that they're a moderately fast typer taking 0.2 seconds per keypress.

Method	GOMS string	Time (seconds)
Ctrl+click	HK MPCR MPCR MPCR MPCR RMKPCR	14.54
Search group	HMC RMPC HMKKKKKKKKKKKKKK KR	9.44

As seen in Table 2 it takes approximately 15 seconds to conduct the grouping of Fitness-Summary and its children using the *ctrl-click* method, while only taking 10 seconds when using the search group method. These results would vary depending on how fast a typer the user was and how good their fine motor skills were. It also depends on how “searchable” the nodes you wish to group are, in cases where the nodes have no common property values it may be faster to select the nodes using the *ctrl-click* method. However in any case where the nodes you want to select share a property with the same value, the search group method is likely to come out on top because multiple nodes can be selected with a single string compared to having the limit of one node been selected by one click.

The results from the GOMS analysis then shows us that there are some cases where having the search method is of considerable benefit and there are some cases where it is not. I did not conduct a more extensive GOMS analysis of different typing speeds and different tasks as it was merely used to show that some benefit can be found from having both methods implemented.

2.2. Challenges

Once a clustering technique is used, whether it be manual or automatic, the issue of naming becomes apparent. Namely, once a composite node is created it must be given a name. In an early version of the ProvOwl system nodes were given short random alpha-numeric names. We found that this soon made the graph incomprehensible and relied on the user manually renaming each node (described in Section 3.1.2) in order for the graph to be usable.

Ideally an interface would give clusters a name describing its contents. I found this is a much harder problem than it at first seems and requires domain level knowledge as well as models of the user and what information is important to them. For example, if the nodes sunflower, daffodil and poppy were to be clustered, it would be useful to name the cluster *plants*. However if the user was a florist this may be too broad a definition and the label *flowers* would be more appropriate.

In some fields this problem has been solved by limiting the number of unknown things. For example, if you create a folder of apps on an iPhone it is automatically named with a label appropriate to the applications inside of it. A folder full of photography apps may be labelled *Photography*. However this differs from the problem we are solving in a variety of ways. Firstly the number of possible labels is limited to the categories in the Apple app store, this creates a finite set of possible label options. Secondly, when an application is uploaded to the app store the author publishes metadata with it such as the category of the app. Using this metadata makes it much easier to name clustered items. Having the same level of metadata in provenance files could be accomplished by having tags associated with each node, these tags could then be used to effectively name a cluster simply by selecting the most used tag from the group of nodes to be clustered. However there is currently no tagging system for provenance nodes.

Our currently, still simple approach uses the name of the node in the cluster closest to the root with the text “group” appended to the end as seen in Figures 2 and 3. While this does not always create the most effective names for clusters, it gives the user a point of reference when looking at clusters which in most cases is enough to trigger the user to what the contents of the cluster is. During usability studying users never renamed nodes unless they specifically wanted to convey information to another user.

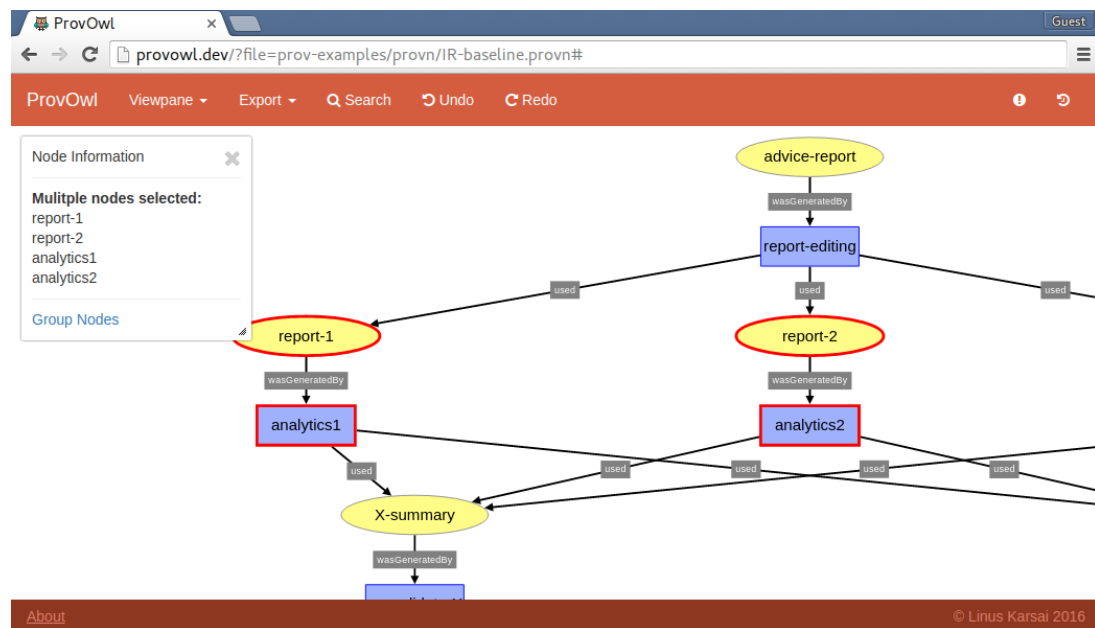


FIGURE 2. Using the *IR-baseline* example the user has selected four nodes for grouping, essentially those related to report-1 and report-2.

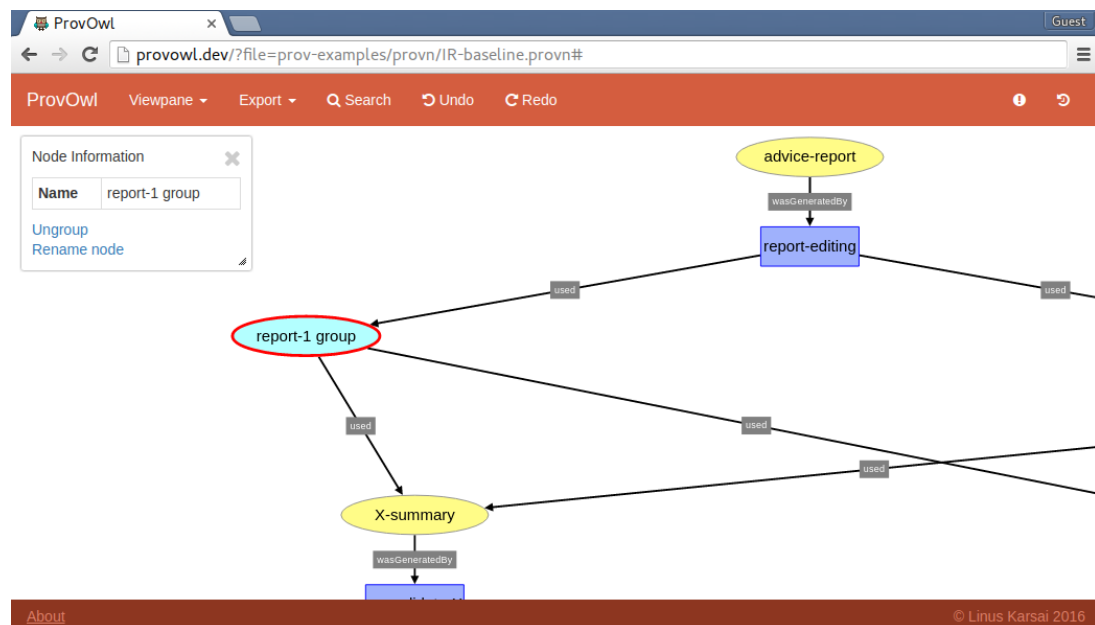


FIGURE 3. When clustering the four nodes selected in Figure 2 ProvOwl creates a composite node using the name of the node closest to the root (in this case breaking a tie by alphabetical order).

CHAPTER 3

Working Prototype

In order to assess the effectiveness of the clustering techniques described above I created a prototype application that implemented them. This prototype was then used in usability studies to assess the effectiveness and usability of the clustering techniques. The prototype I created was a web application called ProvOwl, a screenshot of which is shown in Figure 1. It reads provenance in the PROV-N format and renders a directed acyclic graph for users to interact with. ProvOwl is available to play with at ProvOwl.com and the source code is available at github.com/karsai5/ProvOwl. Below I go into detail about the features implemented in the prototype as well as information about the specifics of its implementation.

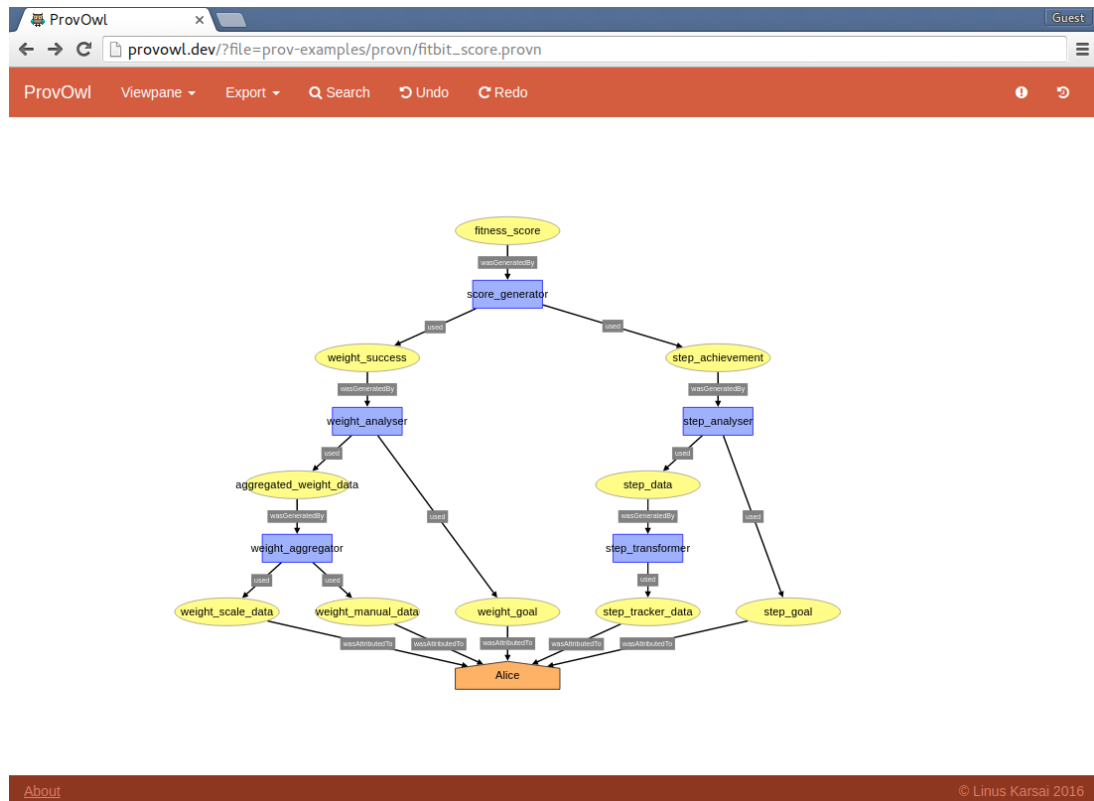


FIGURE 1. A screenshot of ProvOwl rendering the `fitbit_score` prov file used in the usability studies.

3.1. Functions and Design

When creating ProvOwl I quickly discovered that clustering functions could not be implemented in a vacuum and that a usable graph exploration tool would require other features as well. I used Shneiderman’s “The eyes have it: A task by data type taxonomy for information visualizations” [25] as a starting point for creating a usable interface. In his paper he outlines seven usability tasks that should be taken into consideration when creating an advance graphical user interface. These where: *Overview*: a user should be able to view the entire entity been visualised, this is important in giving them a sense of location and scale. *Zoom*: When a user finds a point of interest they should be able to zoom in to view it in more detail. *Filter*: Users should be able to limit what they see on screen to remove information they do not find useful. *Details-on-demand*: A user should be able to view details about elements when they want. But the details should not be constantly on screen and cluttering the view. *Relate*: Information between items should be visible; How information flows from one element to another. *History*: Allow users to be able to undo and replay their actions, this is important in reassuring them that their actions are not permanent and they are free to make mistakes. *Extract*: Once a user has found something interesting they should be able to extract their queries for later exploration or sharing.

Using these usability tasks as a starting point I implemented six main features into ProvOwl: *Movement and rearranging*, *details panel*, *clustering*, *undo*, *sharing* and *click tracking*. Each feature is described below as well as the reasoning for implementing it and its relations to Shneiderman’s seven tasks.

3.1.1. Movement and Rearranging. On first opening a provenance graph, the viewport is positioned to fit the entire graph on screen as seen in Figure 1. This accomplishes Shneiderman’s *overview* task and allows the user to gain an overview of the provenance. Users can then pan around by clicking and dragging. Zooming is accomplished by pressing ctrl+[+,-] or by using the scroll wheel (accomplishing the *zoom* task).

By default the graph’s overall layout is determined using a JavaScript library called dagre¹ that generates layouts for directed acyclic graphs client side. The main skeleton of the algorithm implemented to accomplish this comes from the paper “A Technique for Drawing Directed Graphs” by Gansner et al. [13]. In early versions of the prototype other layout options where available such as circle and breadth-first, but this confused users and where later removed in favour of using dagre exclusively for layout.

Although initial placement is determined using the dagre algorithm, users can also re-arrange nodes manually by clicking and dragging a node. This is different from

¹Dagre javascript library for determining the layout of directed acyclic graphs: <https://github.com/cpetitt/dagre>

clicking and dragging on the background which allows users to pan, however users did not seem to have any difficulty differentiating between the two.

3.1.2. Details panel. When a user selects a single or group of nodes the details panel is shown, as seen in the top left of Figure 2 with the header “Node information”. This lists the attributes and properties of a node² as per Shneiderman’s *details-on-demand* task. At the bottom of the panel in blue text is the list of contextual hyperlinks (as seen in Figure 1 and 2) that enable functions such as renaming and in the case of multiple selected nodes, grouping. Having this panel only appear when nodes are selected means it is not using up space when not displaying information.

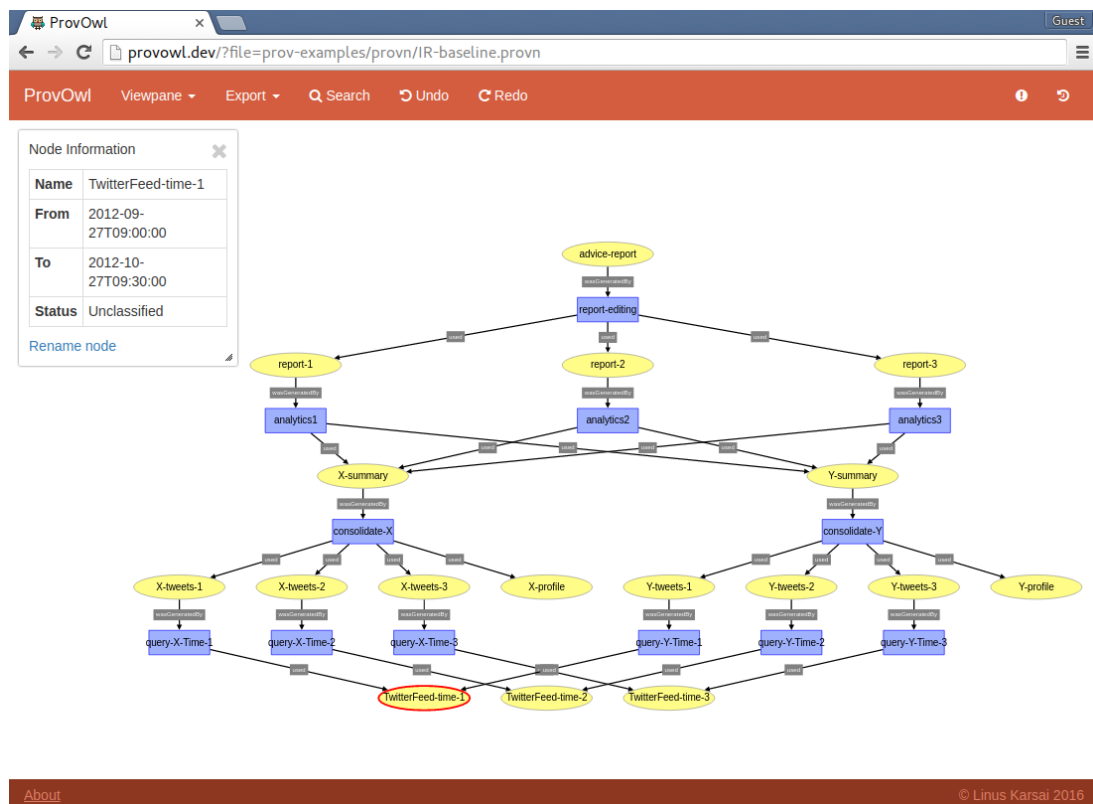


FIGURE 2. A screenshot of ProvOwl with the IR-baseline prov file open. A indicated by the red outline, the node `TwitterFeed-time-1` has been selected and details about it such as name and classification status are shown in the details panel in the top left.

3.1.3. Clustering. As mentioned in Section 2.1 I implemented two ways of clustering nodes, *click+ctrl* and via the search function. In both cases once the user clicks the group link, the currently selected nodes are animated to their new position (the

²Note that both attributes and properties are referred to when discussing nodes although I sometimes use the terms interchangeably they are intrinsically different things. Attributes are data related to the semantics of the node, such as its id and name, whilst properties are key value pairs that can be arbitrarily added to nodes. In most cases there is no reason to treat them differently.

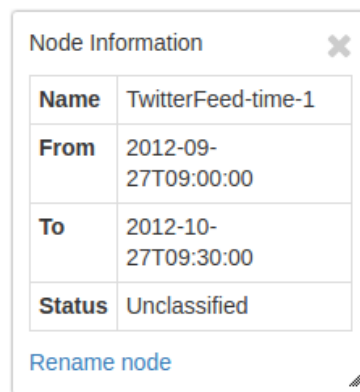


FIGURE 3. A close up of the details panel from Figure 2. This panel shows the attributes and properties of a node as well as a list of contextual actions available in blue links (such as rename, group or ungroup).

node closest to root) and replaced by a composite node, represented by a light blue oval.

To manually cluster nodes a user first selects multiple nodes at once by clicking on each whilst holding down *ctrl*. Once multiple nodes have been selected the user can group them together either by pressing *ctrl*+*g* or by selecting the “group nodes” hyperlink, as seen in the details panel of Figure 4.

The above method is great for a small number of nodes but less effective if a user is trying to cluster more than a dozen nodes, hence the search method of clustering was also implemented. By selecting search from the toolbar the search panel is shown. The search function takes the value entered by the user and runs a regex match on each property of each node. If any property of a node matches the regex it is selected (as indicated by a red outline).

In Figure 4 you can see an example of this. On the top right of the screen is the search panel with the input `X-tweets|query-X-Time`. This query is run on all the nodes and because of the pipe (`|`) character it selects all nodes that are either named `X-tweets-*` or `query-X-Time-*`. Once these nodes are highlighted (as indicated by the red outline) they can be clustered into a composite node by pressing enter, pressing *ctrl*+*g* or clicking “group nodes”. This results in the selected nodes been animated to their new position and replaced by a composite node as seen in Figure 5.

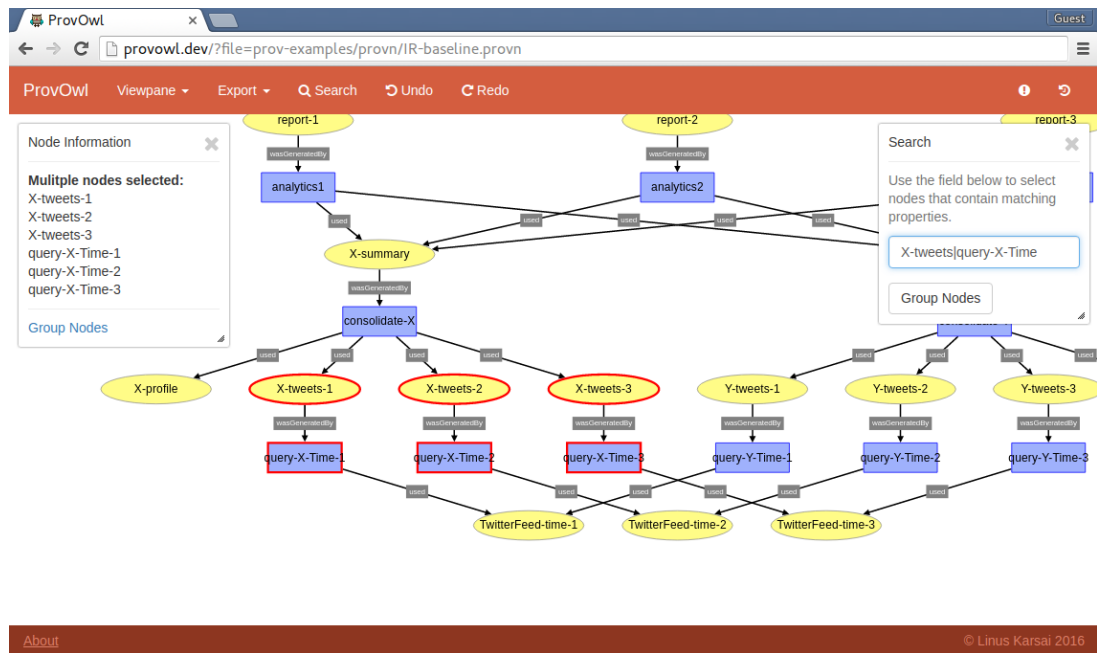


FIGURE 4. A screenshot of ProvOwl with the *IR-baseline.prov* file open; an abstract example that shows the provenance of an advice report presenting analytical information about twitter feeds. The search panel has been used to select a subset of the nodes.

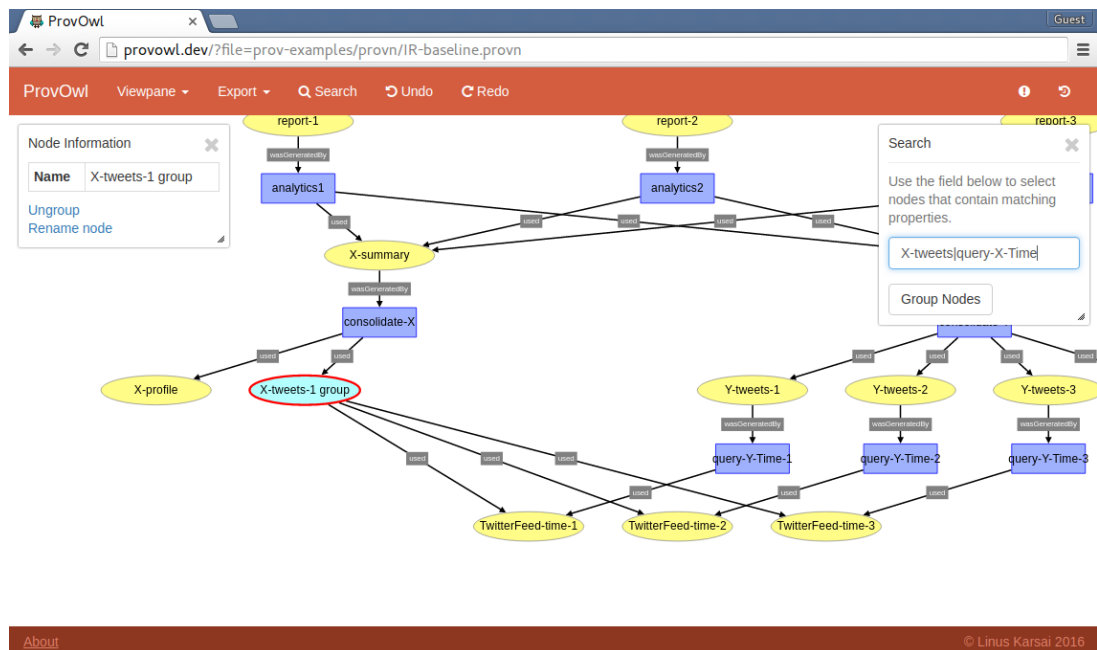


FIGURE 5. Using the *Group Nodes* button in the search panel, the nodes highlighted in Figure 4 have been clustered into a single composite node labelled *X-tweets-1 group*.

3.1.4. Undo/Redo. Having the ability to undo and redo actions is an implementation of Shneiderman’s *history* task and I believe it to be one of the most impactful features for ensuring ease of use. Allowing users to undo and redo their actions gives them a sense that their changes are not permanent and allows exploration of features and the graph without fear of causing irreparable damage. ProvOwl tracks the movement and clustering of nodes. The undo and redo buttons in the toolbar allow users to step backwards and forwards through these actions. Using the history icon in the top right corner of the interface toggles a history pane (as seen in Figure 6) that shows the user what step they are currently at.

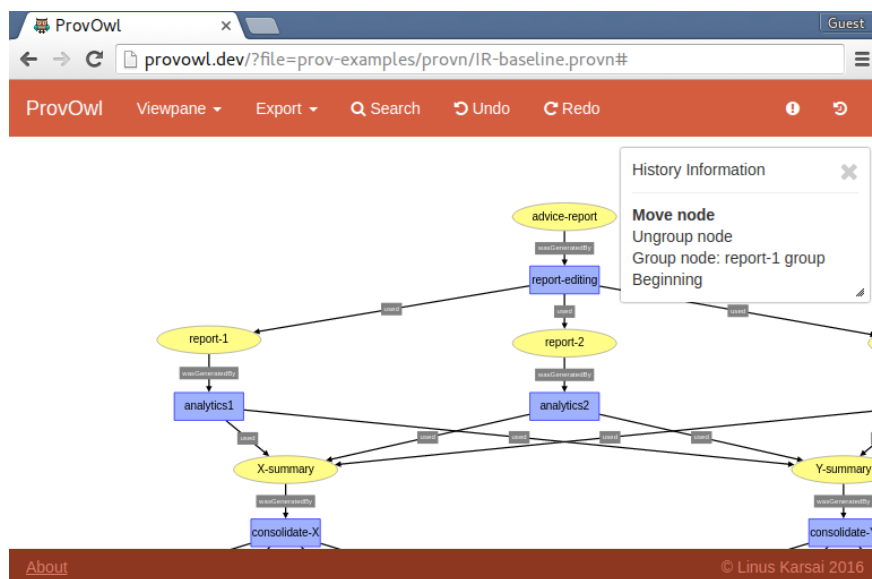


FIGURE 6. In the top right of the interface can be seen the history panel. This indicates what step of history the user is currently at (with bold text). The user can use the undo and redo commands in the toolbar to move backwards and forwards.

3.1.5. Sharing. Once a user has explored a graph and found a point of interest they should be able to save their progress. As described by Shneiderman while describing the *extract* task, this allows further exploration at a later date and the ability to share points of interest with other people.

In the toolbar is the export command. This dropdown menu allows the user to either save the entire graph or the current viewport as an image. The option to export just the viewport allows a user to focus on a certain section. In the future it would also be useful to export a PROV-N file that contained the cluster information.

3.1.6. Click tracking. To facilitate with usability testing and by recommendation of Judy Kay I implemented a click tracker. This logs clicks that the user makes, capturing time, location and which element the user clicked on. This can then be downloaded via the export command as a CSV file. You can see below an example of the csv file generated from exploring a provenance file. This feature doesn’t relate

to usability and therefore doesn't address any of Shneiderman's tasks, instead it is intended to be used as a tool for usability analysis.

	Timestamp	Desc	Element	X Position	Y Position
1	1465352349411	Moved node	ir:X-tweets-3	870	476
2	1465352349471	Clicked node	ir:X-tweets-3	870	476
3	1465352349927	Moved node	ir:X-tweets-2	645	464
4	1465352349999	Clicked node	ir:X-tweets-2	640	464
5	1465352350310	Moved node	ir:X-tweets-1	395	460
6	1465352350349	Clicked node	ir:X-tweets-1	395	460
7	1465352351271	Grouped Nodes ir:X-tweets-1,ir:X-tweets-2,ir:X-tweets-3			
8	1465352360132	Grouped with simple filter		Y	
9	1465352361542	Moved node	-286775817	1510	289
10	1465352362383	Clicked background		1608	576
11	1465352363671	Moved node	-1353429745	298	372
12	1465352363725	Clicked node	-1353429745	298	372
13	1465352363792	Moved node	-1353429745	298	372
14	1465352363902	Clicked node	-1353429745	298	372
15	1465352363915	Clicked node: double click		-1353429745	
16		298 372			
17	1465352365551	Clicked background		235	253
18	1465352365918	Moved node	ir:X-tweets-1	274	359
19	1465352365981	Clicked node	ir:X-tweets-1	274	359
20	1465352366342	Clicked background		430	353
21	1465352366805	Moved node	ir:X-tweets-2	574	376
22	1465352366878	Clicked node	ir:X-tweets-2	573	376
23	1465352367302	Moved node	ir:X-tweets-1	318	378
24	1465352367374	Clicked node	ir:X-tweets-1	318	378
25	1465352367878	Moved node	ir:X-tweets-3	766	366
26	1465352367933	Clicked node	ir:X-tweets-3	766	366
27	1465352368799	Grouped Nodes ir:X-tweets-1,ir:X-tweets-2,ir:X-tweets-3			
28	1465352369935	Clicked background		447	297
29	1465352373590	Moved node	ir:X-summary	620	515
30	1465352376198	Moved node	ir:X-profile	841	664
31	1465352377742	Clicked background		674	619

FIGURE 7. The CSV file generated from a user exploring the IR-baseline provenance file. Each line represent a single click by the user, a small description of the action they completed, what element it effected and the X,Y position of the click (if appropriate).

3.2. Implementation

3.2.1. Web Technologies. I implemented my prototype as a web application. The criteria for picking the web platform was that of accessibility to users and speed of development. Using this criteria I decided to make ProvOwl a web application as its only entry barrier to users is that of a modern web browser. I took the Prov-O-Vis [17] system as inspiration as it was the most accessible of the available provenance

visualisers. Web applications are a field that I have experience with and can quickly develop in.

There were a few pitfalls to this approach that can be addressed through alternative implementations. Because my application is primarily a prototype for testing the usability of manual clustering implementations, the issue of scale is not one that is addressed. If a web application was used for larger provenance files the application may run out of resources in these cases further testing and alternative approaches may be required.

An alternative to a web application would be to write ProvOwl in an OS independent language such as Java or Python; this would allow wide accessibility and more fine grained control of machine resources. However if an OS level language was chosen issues could later occur in presenting provenance on mobile devices and a mobile version of the interface may need to be written. Alternatively if an OS dependant language such as Apple's swift or Microsoft's C# was used features and code would have to be replicated for across the implementations in different languages.

The standard technology for data visualisation in a browser is the D3.js³ JavaScript library. However as mentioned earlier, I used Cytoscape.js to implement my interface because it is specifically focused on graph theory and has graph related functions built in (such as distance algorithms).

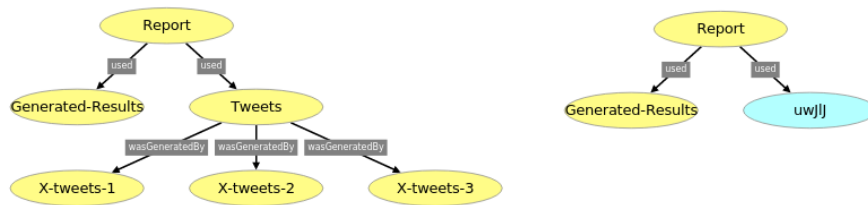
3.2.2. Client Side Processing. When first outlining features of the application I wanted it to be primarily stand alone. Because of this I created PovOwl so that when a provenance file is selected its loaded and rendered client side. This means that even though it run from a web server, no client side computation is required, it can be just as easily run locally on a computer after cloning from the git repo and running a python web server⁴.

3.2.3. Node Clustering. Creating a clustering function in Cytoscape.js required some time and effort because it currently does not have a clustering feature, so I was required to write one from scratch. The goal of this function was to have grouping and ungrouping nodes be non-destructive: if a series of nodes were grouped and ungrouped the resulting graph should be isomorphic to the original.

Cytoscape.js has a "remove" function that will hide a node. Hidden nodes can then be 'restored' into their original position, unfortunately any edges associated with a removed node are permanently destroyed. It was necessary then to store the edges as a data field in the composite node so that when restoring nodes, the original edges could also be recreated.

³D3: Data Driven Documents <https://d3js.org/>

⁴Python's SimpleHTTPServer command can be used to get a web server up and running in a directory, this is described in more detail in this web article: <http://www.linuxjournal.com/content/tech-tip-really-simple-http-server-python>



(A) A provenance graph showing information about a report given on twitter data.

(B) The nodes Tweets and X-tweets-[1-3] have been clustered into the composite node uwJIJ

FIGURE 8. A provenance that has had a subset of its nodes clustered. In an early version of the prototype composite nodes were given random alphanumeric names as seen on the right.

Described below is the pseudocode to create a new cluster. It creates a new node and stores the selected nodes and their edges as properties of it. Then new edges are created to replace the edges that originally linked external nodes to the clustered nodes. Finally the selected nodes are “removed” so that they are no longer visible.

```

1  // Create composite node
2  Create new CompositeNode
3  CompositeNode.originalEdges = All edges in
   neighbourhood of selected nodes // save edges
4  CompositeNode.originalNodes = All nodes that where
   grouped // save nodes
5
6  // Create new edges connected to the composite node
7  for each edge in neighbourhood {
8      if (edge IS NOT internal to group) {
9          Create new Edge({from: externalNode,
10                           to:compositeNode})
11      }
12  }
13  // Remove original nodes
14  Remove selected nodes

```

FIGURE 9. Pseudocode of creating a composite node

However the above code only works in a limited scenario, particularly when ungrouping nodes in the opposite order of creation. For example if you created composite nodes *A*, *B*, *C* (chronologically and with neighbouring edges) you would have to ungroup them in the following order: *C*, *B*, *A*. If you where to ungroup node *A* first, when ungrouping *C* it would try to restore edges to the now non-existent composite node *A*, there is an example of this in Figure 10. In order to fix this I created a class called *GroupManager* to monitor nodes in clusters.

The Group Manager class is the primary way of keeping track of nodes that are hidden inside composite nodes. It contains a tree data structure that references all the groups as well as their child nodes/groups. When unclustering nodes that have edges to nodes not currently in the graph, this class is called and the parent cluster of the node requested, then a new temporary edge can be created linking the composite node and the original edge is stored in an object called *hanging edges* until it can be restored. Every time a cluster is created or destroyed the *hanging edges* object is queried to see if any original edges can be recreated, as seen in Figure 11.

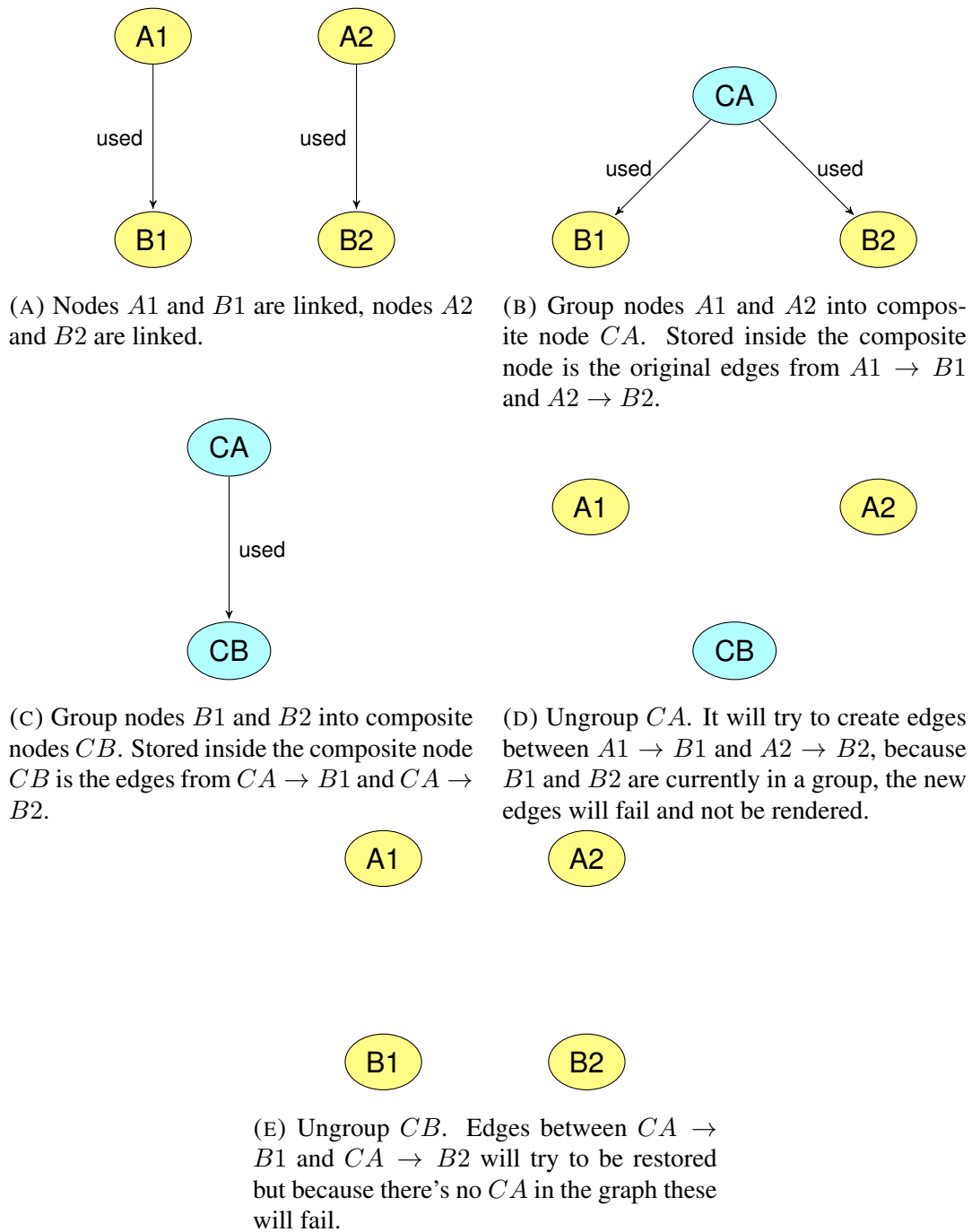


FIGURE 10. An example of issues that occur when trying to ungroup nodes in the incorrect order without the *Group Manager*. As seen in the final step, the original edges connecting the nodes are lost.

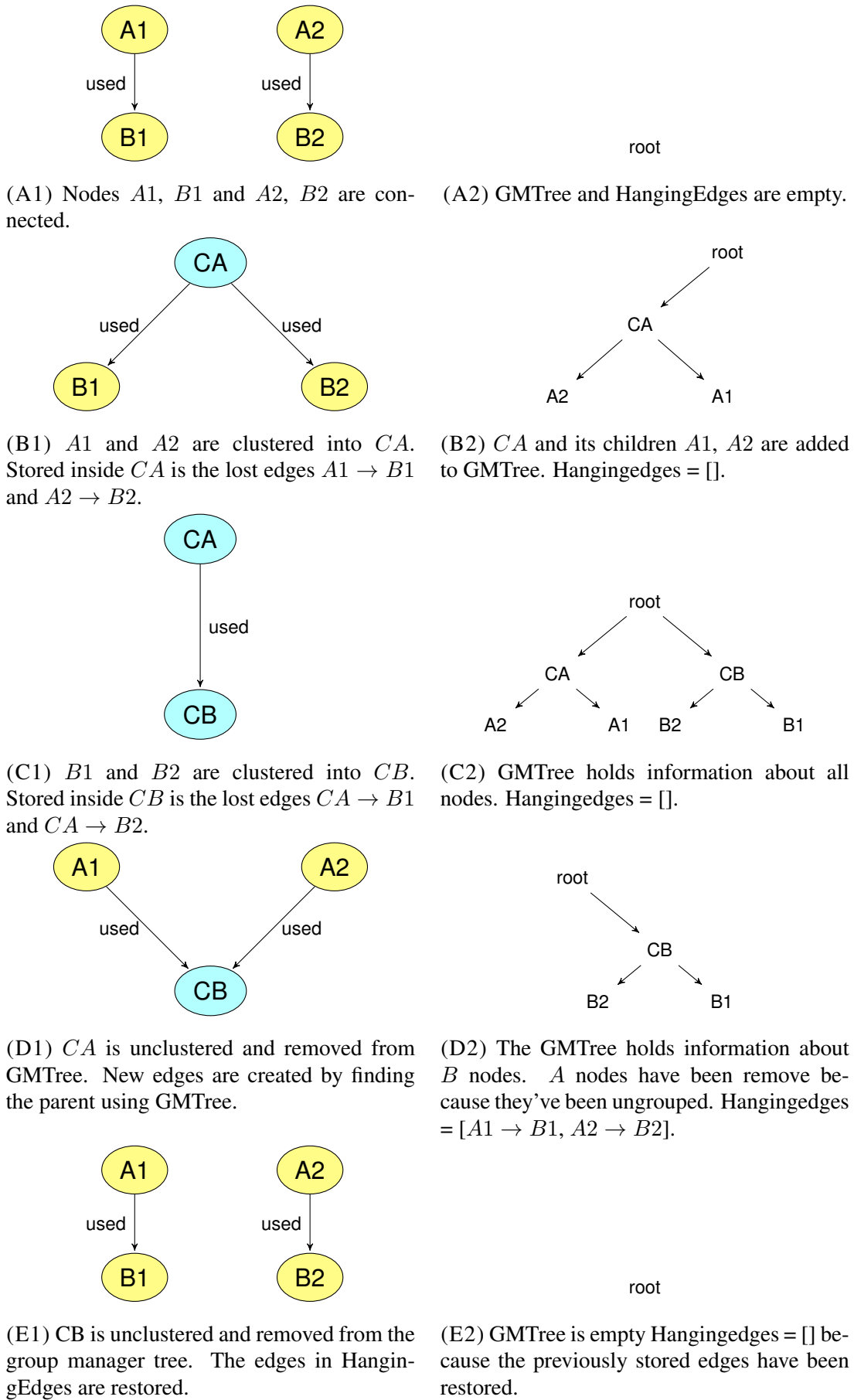


FIGURE 11. The same example as in Figure 10 except this time using the group manager and hanging edges. The group manager stores a tree to track groups and internal nodes as seen on the right hand side. Hanging edges stores a list of edges that could not be restored because one or more connected nodes were missing. After each clustering event an attempt is made at restoring hanging edges.

CHAPTER 4

Usability study

In this paper I have so far described my first contribution: a design for manually clustering nodes and my second contribution a working prototype that implements those designs. My final contribution is a usability study to evaluate learnability, ease of use, efficiency, recovery from errors and user attitude towards my prototype. I particularly felt that this was a requirement for designing an interface because I believe that unless an interface is usable it can not be useful. In parallel to this, the usability study allowed me to evaluate the claim that the clustering methods I describe (*ctrl+click* and search) make for a usable system to cluster provenance. Below I go into details about the design of the study then in the next section I discuss the results.

4.1. Study Design

I decided to conduct a think aloud study with participants while using the ProvOwl application. Think aloud studies are considered the gold standard for testing a prototype and quite simply involves having a user try to accomplish a set of tasks whilst you watch and take notes. Jakob Nielsen, a researcher specialising in discount usability engineering, describes it as the following:

“In a thinking aloud test, you ask test participants to use the system while continuously thinking out loud — that is, simply verbalizing their thoughts as they move through the user interface.”

Participants are encouraged to “think aloud” as they complete tasks to give incite into what they are thinking. I selected this approach for a number of reasons. Firstly, it has been used successfully for a long time. Nielsen describes it as “...the single most valuable usability engineering method.” in his 1993 book *Usability engineering* [21]. Secondly, it requires a small number of users to get meaningful results, a post written in 2012 by Nielsen describes why you only need to test with five users¹. Lastly, I have conducted think aloud studies previously, so they are a natural first thought to me when gauging how usable an interface is. After completing the thing aloud I also had users complete a system usability scale questionnaire about the interface. Similar to think alouds, the SUS questionnaire is one of the most widely used questionnaires in the field of usability because of its simplicity and validity (ability to differentiate a usable interface from an unusable one).

¹Thinking Aloud: The #1 Usability Tool: <https://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool/>

Below I discuss the general format of the interviews and then in detail each of the scenarios users undertook and the tasks associated with each. The entire questionnaire I used for conducting the interviews can be found at Appendix B.

4.1.1. Interview format. The interviews were undertaken wherever was most convenient to the participant, this included at my desk, at the participants house or at coffee shops. Participants were asked a series of background questions, then undertook three scenarios in a think aloud study and finally were asked to complete a system usability scale questionnaire.

4.1.1.1. *Background Information.* General background information was collected about each participant. Gender, age group and highest level of education. These were recorded in order to identify any demographic skew that I may have in my cohort as well as in case any unexpected correlations occurred. The last question in this background survey asked if the participant had heard of the term *provenance* before, and if so to explain what it meant. Because provenance is used in other fields (such as accounting and art dealership) I wanted to identify anyone who had previous concepts of provenance and lineage to identify if this effected their ability to interpret provenance graphs.

4.1.1.2. *Scenarios.* After the participant had answered the general background questions they were given a information sheet that explained the concept of provenance and showed an example of a provenance graph, explaining what each of the different coloured and shaped nodes meant (Appendix A). This allowed participants to have a uniform understanding of the concept of provenance regardless of whether they had heard of it before. Notes were taken of any questions the participant had whilst reading the information sheet.

They were then given my laptop. If the participant was not used to using a touchpad they were also supplied with a USB mouse. Open in a full screen browser was three tabs, one for each scenario the participant had to complete. Each scenario then has a series of tasks they had to either answer or complete, in some cases users required prompting to complete a task, in these cases I noted that the task was completed but with help. Each of the scenarios and tasks are discussed in detail in Section 4.1.2.

4.1.1.3. *SUS Questionnaire.* The System Usability Scale has been around since 1996 when it was described by Brooke in the paper “SUS-A quick and dirty usability scale” [8] and provides a quick easy way of identifying if an interface is easy to use or not. It consists of 10 questions each requiring a response on a scale of 1–5. Users are asked to reply with what they intuitively feel rather than thinking about the questions for an extended period.

I decided to use the SUS questionnaire because it is easy to administer (takes less than five minutes for users to complete), is valid in differentiating between usable and

unusable interfaces [22] and can be used on a small sample size with reliable results². The user was asked to complete the survey after completing the think aloud.

4.1.1.4. *Coded questions.* Each question was given a set of points, user actions that could be checked off if a user did them. For example, in question one of exercise one, the participant was asked *What information is used to calculate your [fitness] score?*. One of the points for this question was *ungrouped nodes*, so if the user ungrouped nodes while answering the question this point was ticked off. Whilst hand written notes were also taken during interviews, these check boxes made for an easy way of identifying what common actions users undertook to complete exercises.

Each point was assigned a code identifying what class of activity it belonged to. In the above example the ungrouping point had the code *grouping-ungroup* because it's completing a grouping function and in particular an ungrouping function.

Each of the codes are listed in Table 1 (page 74) with an explanation of the user behaviour they map to.

4.1.2. Scenarios. The user was asked to undertake three different scenarios, each with its own provenance file and a set of related tasks. Using multiple scenarios allowed me to have tasks that focussed on different objectives. For example the first scenario focuses on provenance understanding whilst the second and third are on provenance modification.

4.1.2.1. *Scenario 1: Alice's Fitness Score.* In the first scenario the user is asked to imagine that they're Alice, they track their fitness information with a Fitbit for steps and a Withings scale for weight (images were shown to describe these objects as seen in Appendix C). The full paragraph given to participants:

Suppose you are Alice. You have been using your FitBit and Withings scales for 3 years to reduce your weight to 61kg and to increase your physical activity to 8000 steps a day. You have just linked your FitBit account to two friends, Bob and Carol. Recently FitBit introduced a fitness score that is shared and ranked with your friends... This makes you wonder just how your score has been calculated.

They were then asked to complete the following tasks, all of which required only a verbal answer.

1. What data about you is used to calculate your score?
2. Identify what aspects of the provenance graph map to your fitness dashboard.
3. What processes does your raw step data go through before been used to calculate the fitbit_score?

²The usability.gov website has a good description of what a SUS questionnaire is and how to use it: <https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html>

4. How do you expect your raw fitbit data would be represented once it got to score_generator?
5. What processes does your Withings scale data go through before been used to calculate the fitbit_score?
6. How do you expect your weight data would be represented once it got to score_generator?
7. Are you concerned about how you data is used to calculate the fitbit score. Explain your reasoning.

The provenance graph they were given to answer these tasks can be seen in Figure 1a. It can be seen from the light blue circles that this included composite nodes. I did this so that users were required to uncluster nodes (either by double clicking or selecting “ungroup” from the contextual commands) in order to properly answer the tasks. This meant that users were introduced to the idea of clustering before having to create them themselves in later tasks. These tasks also allowed the user to get used to reading provenance graphs. Questions 3,4 and 5,6 mirror each other so that if the user had difficulty understanding the way lineage worked in tasks 3,4, requiring prompting, they could then accomplish 5,6 on their own. The last question 7 was simply a way of gauging what users found important, whether they were concerned about privacy or transparency in calculations.

This scenario was purely exploration based and no modifications of the graph were required in order to complete the tasks.

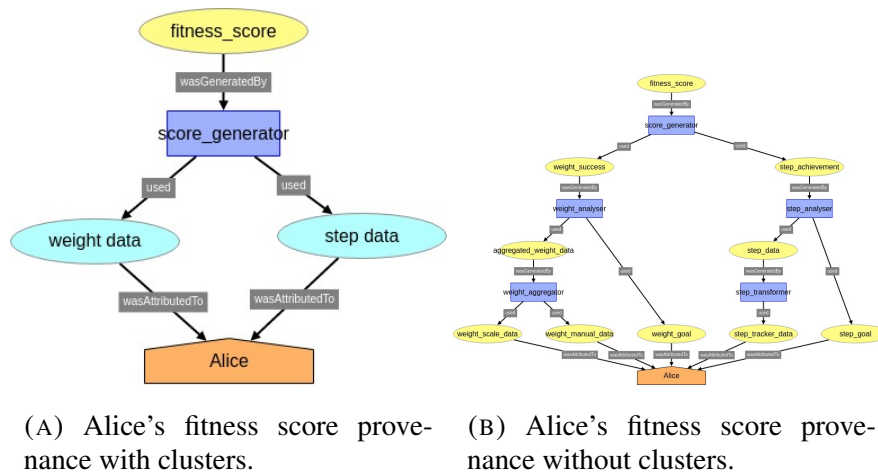


FIGURE 1. The provenance graph used in scenario one showing the lineage of Alice's fitness score. The graph began with clusters as seen on the left. Completely unclustered the graph resembled that on the right.

4.1.2.2. *Scenario 2: Citizen Data Report.* The second exercise involved a larger amount of data and asked participants to imagine themselves as the curator of a citizen science project. Data was been tracked about multiple users and then used in a report that gave feedback on how to improve the cohorts overall fitness. The full paragraph given to participants was:

You are a researcher creating a citizen-data-report that recommends strategies for improving the fitness of your local community. You have step, location, weight and calorie information about members of the community. You are using this information to create multiple reports regarding fitness and weight information which will be used to support a final report addressed to the community on strategies that can be used to improve overall fitness.

They were then asked to compete the following tasks.

1. Describe the graph to me
2. You want to share how the report was generated with a colleague however you want to hide information about the cohort for privacy. Modify the graph in order to hide identifying information about participants. Then save an image of it. Note: try using the filter function for grouping multiple nodes
3. You want to show Alice how her information is been used by sharing an image of the provenance with her that illustrates the lineage of the citizen report and how her data influences it. You do not want Alice to see details about other people in the cohort.

The first task just required that the user describe the graph verbally to me. This did two things: it let me check that the user was correctly interpreting the information stored in the graph as well as forcing users to study and understand the graph (in early tests of the study without this question users would make mistakes later because of poor understanding of the graph).

The second task required the user to cluster nodes and in some cases to rename nodes. This task focused on users *hiding* information. It required 40 nodes to be clustered together so participants sometimes needed prompting towards using the search command.

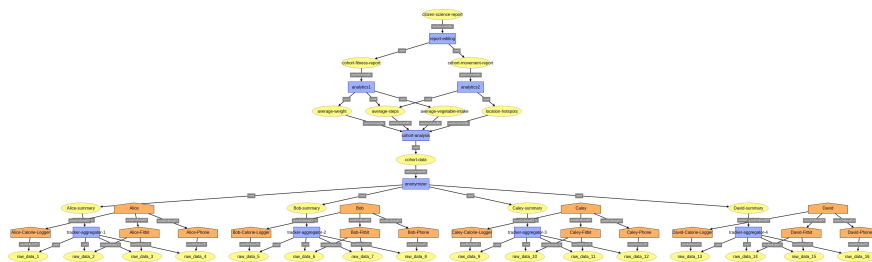


FIGURE 2. The provenance graph used in scenario 2 showing the lineage of a citizen science report. The top section of the graph is related to the report writing process whilst the fan out at the bottom shows information stored about five different people.

4.1.2.3. *Scenario 3: Twitter Data Report.* The third and final scenario was basically a duplicate of the second scenario on a more abstract provenance graph. It also

focused on getting users to convey certain information through the graph rather than just hiding information. The provenance given to the user was that of a advice-report concerning information from multiple twitter feeds. The paragraph given to participants was:

This provenance graph is an abstract representation of a report based on twitter data.

They were then asked to complete the following tasks:

1. Describe the graph to me
2. You want to modify the graph in order to convey the following information to a non-technical user: An advice report was created using data from 3 other reports based on twitter data from two different users.
3. You want to show to user X that their data was used in the advice report by sharing an image of the provenance with them. Modify the graph to best represent this information.

Same as scenario two the first task was used as a way of letting me check the user correctly understood the graph. The second and third tasks are a little different. They focus on asking the user to convey information by simplifying the graph. Compared to scenario two the results of these tasks a much more different from user to user. This seems to be related to different people explaining things different ways, it seemed that people who read graphs a lot were happy to leave the graph (seen in Figure 3) the way it was because they could easily read information from it.

The reasoning behind having this scenario similar to scenario two is that if the user required a large amount of prompting the complete scenario two then this gives them a chance to complete a clustering task without the help of the supervisor. If the user had managed to successfully cluster in scenario two then we got information about how they conveyed information.

4.1.3. Task Coding. To help with analysing the results from the think aloud study the tasks were coded to represent what *conceptual* activity and *concrete* actions the user was expected to perform.

The coding was used to identify what high level activity the user was accomplishing, for example if the task asked to trace the lineage of an entity, this would accomplish the conceptual activity of *identifying lineage*. There was five conceptual tasks the user had to perform to complete all the tasks: *Understanding lineage*: The user had to be able to understand how lineage flowed through a provenance graph. *Un-clustering Nodes*: The user had to uncluster nodes. *Cluster Nodes*: The user had to cluster nodes (via either method). *Hide information*: The user had to hide information from a provenance graph. *Highlight Information*: Hiding information in a provenance graph.

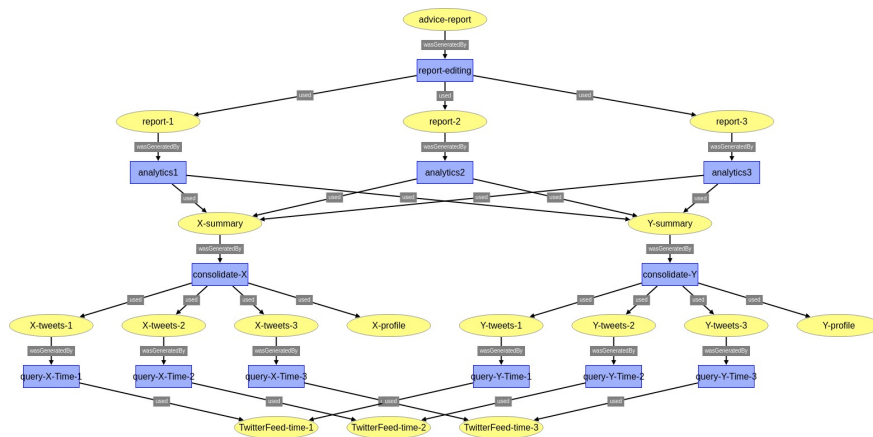


FIGURE 3. The provenance graph used in scenario 3 showing the lineage of a report on twitter data. This is a more abstract provenance graph and shows twitter data from three raw streams been analysed into three reports and finally used in a single *advice-report*.

In Table 1 is a list of each scenario, corresponding tasks and what conceptual activity it accomplishes. From this table we can see that scenario one focuses primarily on ensuring that users conceptually understand the concept of lineage, particularly tracing lineage. It also gives the user a chance to uncluster nodes. Scenarios two and three then focus on hiding nodes and highlighting information, although note that both include a preliminary task that verifies the users understanding of lineage.

TABLE 1. This shows when users were given the opportunity to accomplish different high level concepts.

	Scenario 1							Scenario 2			Scenario 3		
	t1	t2	t3	t4	t5	t6	t7	t1	t2	t3	t1	t2	t3
Understand Lineage	✓	✓	✓	✓	✓	✓		✓			✓		
Un-cluster Nodes	✓		✓	✓	✓	✓							
Cluster Nodes									✓	✓		✓	✓
Hide Information									✓	✓			
Highlight Information									✓	✓		✓	✓

Each task was then given a series of concrete sub tasks that represented things I thought the user would do. For example in task one of scenario one I expected the user would uncluster nodes. Each of these subtasks was also given a concrete code to represent the type of action the user was accomplishing. Using the example before of unclustering nodes, this would have the concrete code of *grouping-ungroup*. A full list of the sub tasks and concrete codes can be found at Appendix F. These subtasks were printed as check boxes on the survey sheet so that I could quickly tick them off as users accomplished them. It meant that in conjunction with qualitative feedback

I also had quantitative results related to how often users completed certain concrete actions (like using part of the interface or following a line of lineage).

4.2. Study Results

I conducted the study on five users. The gender balance was 60% male and 40% female. All participants were between 15 and 44 and they had all completed some level of tertiary education (either a bachelors or masters degree). Of the five people interviewed two of them had heard of the term provenance before, one describing it as “something to do with the history of a file”. Below I discuss the results from the think aloud and then the results from the SUS survey.

4.2.1. Think Alouds. In the first scenario it was required that the participant uncluster nodes in order to view entities required for the tasks. Participants had trouble realising that nodes were hidden inside the composite nodes and three of the users required prompting so that they would uncluster nodes. As users continued through these tasks their ability to read lineage improved. In task 3 “What processes does your raw step data go through before been used to calculate the fitbit_score?” two of the users required prompting and none of the users successfully identified every entity that the step data passes through. However by task 5 “What processes does your Withings scale data go through before been used to calculate the fitbit_score?” users were able to successfully complete the tasks with no prompting and identify all the entities Withings scale data passes through. This suggests that by the end of scenario one users were able to successfully trace lineage through the application. Interestingly the clickdata shows that the most common action in this scenario was moving nodes around even though it was not required to complete any of the tasks. This could be attributed to users playing around with the system, they seemed to enjoy been able to drag nodes around the screen. Most users intuitively realised that double clicking a composite node would uncluster it, this technique was used three times more often than selecting ungroup from the contextual links in the details panel.

The second scenario involved around 50 nodes (compared to 16 for the first). Every participant commented on the size of the graph “Gasp”, “That’s more complicated looking!”. As a lead on from the previous scenario it seemed that participants had a sound understanding of provenance by this stage as all participants identified 80% of the important graph elements when asked to describe the graph. In tasks 2 most users required some sort of prompting in order to cluster nodes, whether by explicitly having the process described to them or just minor points in the right direction. One participant completed the tasks without any help. Users used a combination of *ctrl+click* grouping (80%) and search grouping (40%). Most users (80%) then renamed nodes in order to hide the names of participants. A common mistake made was participants would try to double click a node to rename it, accidentally opening clusters instead. By the third task most users could effectively group without any prompting, with only two of the participants needing help.

In the third scenario all participants completed the first task (describing the graph) without any need for help. Participants also managed to complete tasks two and three without prompting. I did note that some users were happy to leave the graph in its current state for task two so I had to push for them to simplify the graph further, suggesting that a “non-technical” user may not understand it. Some users had an issue where they would hold down shift while clicking on nodes to try select multiple nodes.

Every time users completed a task I scored their success on a scale of 0–3. Where a zero indicated that the participant required a large amount of help completing the task and 3 indicated that they completed the task without any prompting. In Figure 4 we can see the results from these success scores as the mean of all participants. Figure 4 shows that participants initially required a lot of help for the beginning of the first scenario. By the seventh task most users were able to complete tasks without any help. Users then needed help for the beginning of the second and third scenario, however after this initial push in the right direction users quickly became confident in using the application. This graph shows that with no training or background knowledge in the field of provenance, a person can start using the ProvOwl system to effectively create clusters. Users never had to be prompted on how to do the same thing twice, in each of the think alouds hints on how to cluster were only given once. This indicates that users had no usability issues clustering once they found the feature. It is also interesting to note that as was hypothesised in the design stage, users used both clustering methods, *ctrl+click* and search, to complete tasks indicating that both methods are relevant and they both fulfil different needs.

Throughout the study users requested some features they thought would be useful. Its important to note that users may know what they want but they do not know what they need, as Henry Ford said “If I had asked people what they wanted, they would have said faster horses.” So it is important to take user suggested features with a grain of salt. However I do believe the following features suggested would be useful to users. *Box-select*: this would allow users to drag a box on screen while holding shift in order to select nodes. *Cluster children*: this would allow a user to select a node and cluster all child nodes. I also noticed a lot of users would cluster single entity/activity relations together to simplify nodes, this could be implemented in an auto-simplify feature.

4.2.2. System Usability Study Questionnaire. The results from the usability study are divided into two distinct groups. The first group as seen in Table 2 has a very high SUS score. This places the interface at an usability grade of around A-B³, which is the indicator of a very usable interface. Scores of 80 and above are usually the point where users are likely to recommend a product to a friend. These scores could be affected by the backgrounds of the two participants as one had an existing understanding of provenance and the other frequently did data analysis activities such as data mining.

³Notes about interpreting SUS scores can be found in this article: <http://www.measuringu.com/sus.php>

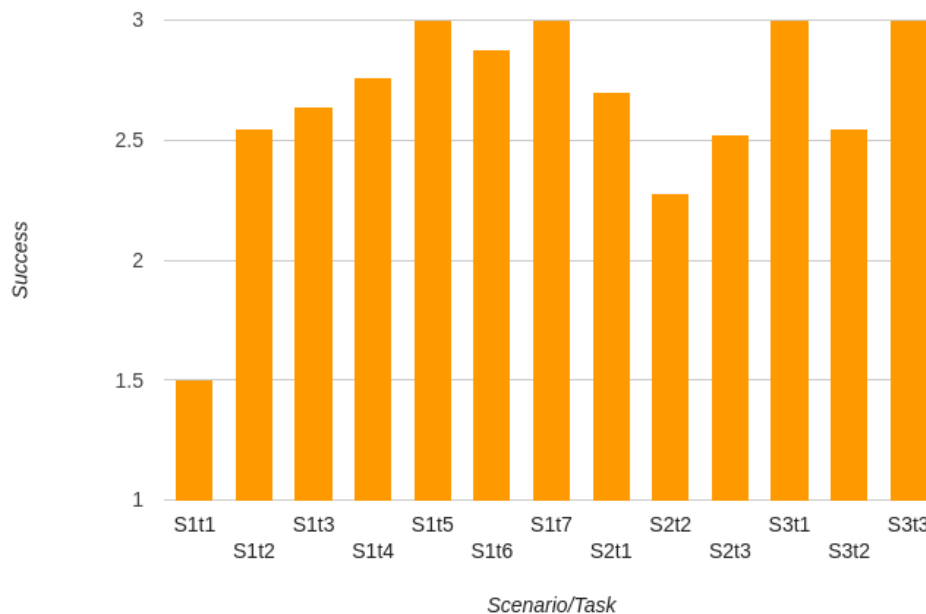


FIGURE 4. This shows the level of help required for a particular scenario/task. If a participant required a lot of help to complete they were given a success score of 0. If they accomplished the task without any help at all they were given a score of 3. The results graphed here is the mean of all participants.

However there is a second group of much lower scores as seen in Table 3, placing the interface at a much lower usability grade. After further analysis of the participants and their interviews it seems that the results may be contributed to lack of sleep in on instance and difficulties interpreting questions in another. User five is the most interesting case because they completed all the tasks without any prompting and still gave the interface an unusually low score of 52.5. I believe this may be because they found some bugs related to how the undo/redo function works and this may have tainted their result.

Overall it is promising to see such high scores from the SUS questionnaire. After implementing some of the user requested features above and fixing the undo/redo bug it would be interesting to see how ProvOwl would score on a larger cohort.

TABLE 2. This higher scoring group can be contributed to the users backgrounds.

User	Score	Notes
1	75	Background in IT research, frequently explores and mines information on a daily basis. Logs information about themselves.
2	85	Strong understanding of digital provenance.

TABLE 3. This lower scoring group seems to be contributed to a number of reasons.

User	Score	Notes
3	40	User was tired (complained about lack of sleep) and sped through the tasks without much thought
4	55	English was a secondary language, participant had difficulty understanding the explanation of provenance, this also affected their ability to interpret questions.
5	52.5	Was very competent in use of the application and completed all tasks without prompting. They found some bugs in ProvOwl perhaps influencing their score.

Conclusions and Discussion

Provenance is a type of metadata representing the lineage of a digital object. In the field of personal data management it can be used to help users understand how their personal data is used, an issue that is particularly relevant as we store more and more information about ourselves. The problem is that raw provenance is too difficult to understand and the current industry standard for visualising it (directed acyclic graphs) is still too complex for non technical users (and in a lot of cases technical users).

To solve this problem I designed two methods for manually clustering nodes, *ctrl+click* and search. Clustering could then be used to simplify provenance graphs in order to hide irrelevant information or focus attention on certain areas of a provenance graph, for example, highlighting how a single users information was used in a report. I then implemented a working prototype so that a usability study could be conducted on the usability and effectiveness of each clustering technique. Using both a think aloud study (the gold standard of usability feedback) and a SUS questionnaire (one of the most widely used questionnaires) I conducted a comprehensive usability study on the ProvOwl application.

The results from the usability study showed that after been prompted to the existence of the clustering function users would use both the *ctrl+click* and search clustering methods in order to simplify graphs. After a small amount of time users were comfortable and confident using the interface to modify graphs to convey meaning, indicating that ProvOwl is a usable interface for clustering graphs.

5.1. Future Work

In the future it would be useful to expand on some of ProvOwl's features as well as implementing some of the other clustering techniques recommended by users in the think aloud study. I would then propose to re-conduct the usability studies and in particular the SUS questionnaire to see if the same divide in results appears. It would also be useful to conduct a separate usability study that focuses on what makes effective clustering when trying to convey information to users, results from this could then be given as feedback to users when manually clustering.

Future challenges involve allowing users to search for all nodes that match one regex but not another. It would also be useful to only match searches on particular properties of a node, for example a user may only want to match their search on *author*

of a node. Many users during usability studying requested the ability to cluster all the children of a node, extending from this it would also be useful to allow user to cluster based on a nodes relationships with other nodes, for example “Cluster all nodes that match the following regex `X-tweets|query-X-Time` and are not derived from `TwitterFeed-time-3`. As mentioned in the think aloud results (4.2.1) it would also be useful to have a feature that lets users draw a box over the nodes they want to cluster.

In conjunction with manual clustering it would also be useful to implement some of the automatic clustering techniques discussed in related work [6, 24]. These automatic techniques used with the manual techniques discussed here could be used to help modify much larger graphs, using automatic clustering to create initial simplification and then manual clustering to fine tune the results.

In the future I would also like to expand on the export abilities of the prov owl application to allow exporting in the PROV-N standard. This would allow sharing of provenance graphs with manually created clusters. This would also include expanding the current provenance standard to include syntax for representing nodes that are clustered together.

Lastly, server side computation may be a useful feature for the future, a server with higher computational power and resources may be able to find and analyse interesting details about a file. This could also be used to explore incredibly large provenance files. By having the graph loaded server side and only sections required sent to the client, the computational load on the client could be greatly reduced.

Bibliography

- [1] James Abello, Frank Van Ham, and Neeraj Krishnan. “ASK-GraphView: A large scale graph visualization system”. In: *IEEE Transactions on Visualization and Computer Graphics*. Vol. 12. 5. 2006, pp. 669–676. ISBN: 1077-2626. DOI: 10.1109/TVCG.2006.120.
- [2] M David Allen et al. “Capturing Provenance in the Wild”. In: *Provenance and Annotation of Data and Processes* (2010), pp. 98–101.
- [3] Khaled Bachour et al. “Provenance for the People : An HCI Perspective on the W3C PROV Standard through an Online Game”. In: *CHI Crossings, Seoul, Korea*. 2015, pp. 2437–2446. ISBN: 9781450331456.
- [4] Khalid Belhajjame et al. “PROV Model Primer”. In: *W3C Working Group Note*. 2013. URL: <http://www.w3.org/TR/2013/NOTE-prov-primer-20130430/>.
- [5] O Biton, S Cohen Boulakia, and S B Davidson. “Zoom*UserViews: Querying Relevant Provenance in Workflow Systems”. In: *33rd international conference on Very large data bases*. 2007, pp. 1366–1369. ISBN: 9781595936493. URL: <http://www.vldb.org/conf/2007/papers/demo/p1366-biton.pdf>.
- [6] Michelle A. Borkin et al. “Evaluation of filesystem provenance visualization tools”. In: *IEEE Transactions on Visualization and Computer Graphics* 19.12 (2013), pp. 2476–2485. ISSN: 10772626. DOI: 10.1109/TVCG.2013.155.
- [7] Uri Braun, Avraham Shinnar, and Margo Seltzer. “Securing Provenance”. In: *Proceedings of the 3rd conference on Hot topics in security (HOTSEC’08)* (2008), pp. 1–5.
- [8] John Brooke et al. “SUS-A quick and dirty usability scale”. In: *Usability evaluation in industry* 189.194 (1996), pp. 4–7.
- [9] S. P Callahan et al. “VisTrails : Visualization meets Data Management”. In: *ACM SIGMOD International Conference on Management of Data* (2006), pp. 745–747. ISSN: 07308078. DOI: 10.1145/1142473.1142574.
- [10] Lucian Carata et al. “A Primer on Provenance”. In: *Queue* 12.3 (2014), pp. 10–23. ISSN: 15427730. DOI: 10.1145/2602649.2602651. URL: <http://dl.acm.org/citation.cfm?doid=2602649.2602651>.
- [11] Stuart K Card, Allen Newell, and Thomas P Moran. “The psychology of human-computer interaction”. In: (1983).
- [12] James Cheney. “A formal framework for provenance security”. In: *Proceedings - IEEE Computer Security Foundations Symposium* (2011), pp. 281–293. ISSN: 19401434. DOI: 10.1109/CSF.2011.26.
- [13] Emden R. Gansner et al. “A Technique for Drawing Directed Graphs”. In: *IEEE Transactions on Software Engineering* 19.3 (1993), pp. 214–230. ISSN: 00985589. DOI: 10.1109/32.221135.

- [14] Paul Groth et al. “Requirements for Provenance on the Web”. In: 7.1 (2012), pp. 39–56. ISSN: 1746-8256. DOI: 10.2218/ijdc.v7i1.213. URL: <http://www.ijdc.net/index.php/ijdc/article/view/203>.
- [15] Pj Guo and M Seltzer. “BURRITO : Wrapping Your Lab Notebook in Computational Infrastructure”. In: *USENIX Workshop on the Theory and Practice of Provenance (TaPP)*. 2012, p. 4. URL: <https://www.usenix.org/system/files/conference/tapp12/tapp12-final10.pdf> <http://pgbovine.net/burrito.html>.
- [16] Ragib Hasan, Radu Sion, and Marianne Winslett. “Preventing history forgery with secure provenance”. In: *ACM Transactions on Storage* 5.4 (2009), pp. 1–43. ISSN: 15533077. DOI: 10.1145/1629080.1629082.
- [17] R Hoekstra and P T Groth. “PROV-O-Viz - Understanding the Role of Activities in Provenance”. In: *5th International Provenance and Annotation Workshop (IPAW)* (2014), pp. 1–6. ISSN: 16113349. DOI: 10.1007/978-3-319-16462-5_18. URL: <http://hdl.handle.net/1871/51388>.
- [18] Peter Macko and M Seltzer. “A general-purpose provenance library”. In: *USENIX Workshop on the Theory and Practice of Provenance (TaPP)*. 2012. URL: <https://www.usenix.org/system/files/conference/tapp12/tapp12-final9.pdf>.
- [19] Kiran-Kumar Muniswamy-Reddy, Peter Macko, and Margo Seltzer. “Provenance for the Cloud”. In: *Proceedings of the 8th USENIX Conference on File and Storage Technologies* (2010), pp. 14–15. URL: <http://dl.acm.org/citation.cfm?id=1855511.1855526> http://www.usenix.org/event/fast10/tech/full%7B%5C_%7Dpapers/muniswamy-reddy.pdf.
- [20] Kiran-Kumar Muniswamy-Reddy et al. “Provenance-aware Storage Systems”. In: *Proceedings of the Annual Conference on USENIX '06 Annual Technical Conference*. ATEC '06. Berkeley, CA, USA: USENIX Association, 2006, p. 4. URL: <http://dl.acm.org/citation.cfm?id=1267359.1267363>.
- [21] Jakob Nielsen. *Usability engineering*. Elsevier, 1994.
- [22] Jeff Sauro. “Measuring usability with the system usability scale (SUS)”. In: (2011).
- [23] Doug Schaffer et al. “Navigating hierarchically clustered networks through fish-eye and full-zoom methods”. In: *ACM Transactions on Computer-Human Interaction* 3.2 (1996), pp. 162–188. ISSN: 10730516. DOI: 10.1145/230562.230577.
- [24] M.I. Seltzer and Peter Macko. “Provenance Map Orbiter: Interactive Exploration of Large Provenance Graphs”. In: *USENIX Workshop on the Theory and Practice of Provenance (TaPP)*. 2011. URL: <http://dash.harvard.edu/handle/1/5168866>.
- [25] Ben Shneiderman. “The eyes have it: A task by data type taxonomy for information visualizations”. In: *IEEE Symposium on Visual Languages*. 1996, pp. 336–343. ISBN: 0818674695. DOI: 10.1109/VL.1996.545307. URL: http://ieeexplore.ieee.org/xpls/abs%7B%5C_%7Dall.jsp?arnumber=545307.

Appendices

Appendix A.	Provenance Primer	54
Appendix B.	Interview Questionnaire	55
Appendix C.	Participant Information Sheet	65
Appendix D.	PROV File: Alice’s Fitness Score	69
Appendix E.	PROV File: IR-Baseline	70
Appendix F.	Concrete codes	74

Appendix A. Provenance Primer

Provenance

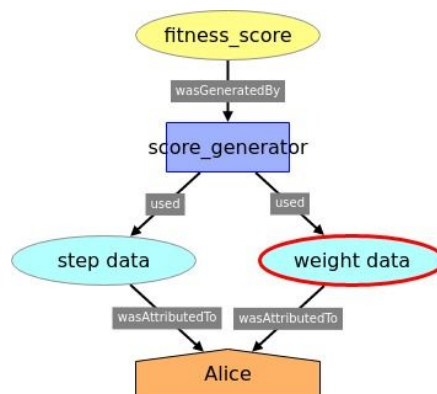
The concept of provenance has been around for a long time, often used in antiques to ensure authenticity and quality of a piece.

In this study, we use the term provenance to refer to digital provenance. Digital provenance is historical information about a project. For example, in exercise one the provenance of fitness score represents what other things and processes created it.

Below is a key of what different coloured and shaped nodes represent in a provenance graph:

- An **orange coloured shape** shows the responsible party. This can be a person but can also be an object such as a fitness tracker.
- **Yellow circles** represent data. It could hold a single value such as *fitness_score*, or rows of data like *step_tracker_data*.
- **Light blue circles** are cluster nodes, this is used when objects are grouped together and represented as a single node. *Weight data* is an example clustered node that contains nodes related to weight (you can double click to open these)
- **Dark blue squares** are processes. For example, *score_generator* is a process that combines step and weight data to make a fitness score.

Note: the arrow notation may seem counter intuitive at first. In the example below the arrow between *weight_data* and *score_generator* means that *score_generator* used information from *weight_data*. Arrows point further into the past.



Appendix B. Interview Questionnaire

Linus Karsai
Thesis Interviews

This questionnaire is automatically read by a computer program. Please use a pen for filling in your answers.

Check: ☒ You can check any number of boxes in selection questions.

Uncheck to correct: ☐ For questions with a range (1–5) choose the answer the mark that fits best.

Thank you for agreeing to be a part of my usability study. It will be split into three parts:

- General background info
- Three exercises
- Simple survey

1 Background Information

1.1 Gender

☐ Male ☐ Female ☐ Other

1.2 Age Group

☐ 15-24
☐ 25-44
☐ 55-64
☐ 65+

1.3 Highest level of education

☐ High School ☐ Diploma ☐ Bachelor Degree


☐ Masters Degree Other:


1.4 Have you heard of the concept of provenance before? If so, explain it in your own words.

☐ No

Yes:

1.5 Prov info sheet notes:


NONE


2230979588 0001

2 Exercise 1

Suppose you are Alice. You have been using your FitBit and Withings scales for 3 years to reduce your weight to 61kg and to increase your physical activity to 8000 steps a day. You have just linked your FitBit account to two friends, Bob and Carol. Recently FitBit introduced a fitness score that is shared and ranked with your friends... This makes you wonder just how your score has been calculated.

2.1 Exercise 1 general notes:

2.2 What data about you is used to calculate the score?

- | | | |
|---|---|--|
| <input type="checkbox"/> weight_scale_data | <input type="checkbox"/> weight_manual_data | <input type="checkbox"/> weight_goal |
| <input type="checkbox"/> step_tracker_data | <input type="checkbox"/> step_goal | <input type="checkbox"/> question alice was attributed to fitbit_score |
| <input type="checkbox"/> notice manual and automatic logging was used | <input type="checkbox"/> ungrouped nodes | |

2.3 Success

☐ ☐ ☐ ☐ ☐

2.4 Notes:

2.5 Identify what aspects of the provenance graph map to your fitness dashboard.

- | | | |
|--|------------------------------------|--|
| <input type="checkbox"/> raw_scale | <input type="checkbox"/> raw_steps | <input type="checkbox"/> fitness_score |
| <input type="checkbox"/> ungrouped nodes | | |

2.6 Success

☐ ☐ ☐ ☐ ☐

NONE



2230979588 0002

2.7 Notes:

2.8 What processes does your raw fitbit data go through before been used to calculate the fitbit_score?

- | | | |
|---|--|--|
| <input type="checkbox"/> step_transformer | <input type="checkbox"/> step_data | <input type="checkbox"/> step_analyser |
| <input type="checkbox"/> step_achievement | <input type="checkbox"/> score_generator | <input type="checkbox"/> fitness_score |
| <input type="checkbox"/> ungrouped nodes | | |

2.9 Success

☐ ☒ ☐ ☐ ☐

2.10 Notes:

2.11 How do you expect your raw fitbit data would be represented once it got to score_generator?

- | | | |
|--|-------------------------------------|---|
| <input type="checkbox"/> understands data transformation | <input type="checkbox"/> not at all | <input type="checkbox"/> in anonymised form |
| <input type="checkbox"/> ungrouped nodes | | |

2.12 Success

☐ ☐ ☐ ☐ ☐



NONE



2230979588 0003

2.13 Notes:

2.14 What processes does your Withings scale data go through before been used to calculate the fitbit _score?

- ☐ aggregated ☐ analysed ☐ algorithm to determine sucess
☐ ungrouped nodes ☐ identified manual and auto-
matic input

2.15 Success

☐ ☐ ☐ ☐ ☐

2.16 Notes:

2.17 How do you expect your weight data would be represented once it got to score _generator?

- ☐ understands data transforma- ☐ not at all ☐ in anonymised form
tion
☐ ungrouped nodes

2.18 Success

☐ ☐ ☐ ☐ ☐



NONE



2230979588 0004

2.19 Notes:

2.20 Are you concerned about how your data is used to calculate the fitbit score. Explain your reasoning.

☐ Worried about weighting

☐ Worried about privacy

☐ Not-concerned

2.21 Success

☐ ☐ ☐ ☐ ☐

2.22 Notes:



NONE



2230979588 0005

3 Exercise 2

You are a researcher creating a citizen-data-report that recommends strategies for improving the fitness of your local community. You have step, location, weight and calorie information about members of the community. You are using this information to create multiple reports regarding fitness and weight information which will be used to support a final report addressed to the community on strategies that can be used to improve overall fitness.

3.1 Exercise 2 general notes:

3.2 Describe the graph to me.

- | | | |
|---|--|--|
| <input type="checkbox"/> people with identical data | <input type="checkbox"/> cohort data anonimised | <input type="checkbox"/> averages created from anonymised cohort |
| <input type="checkbox"/> two reports created | <input type="checkbox"/> citizen-science-report based on two reports | <input type="checkbox"/> silmple filter |
| <input type="checkbox"/> groupin manually | | |

3.3 Success

☒ ☐ ☐ ☐ ☐

3.4 Notes:

3.5 You want to share how the report was generated with a colleague however you want to hide information about the cohort for privacy. Modify the graph in order to hide identifying information about participants.

- | | | |
|---|--|---|
| <input type="checkbox"/> Group nodes | <input type="checkbox"/> ungroup nodes | <input type="checkbox"/> silmple filter |
| <input type="checkbox"/> rename nodes | <input type="checkbox"/> export image | <input type="checkbox"/> screenshot image |
| <input type="checkbox"/> groupin manually | | |

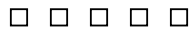


NONE



2230979588 0006

3.6 Success



3.7 Notes:



3.8 You want to show Alice how her information is been used by sharing an image of the provenance with her that illustrates the lineage of the citizen report and how her data influences it. You don't want Alice to see details about other people in the cohort.

- | | | |
|---|--|---|
| <input type="checkbox"/> Group nodes | <input type="checkbox"/> ungroup nodes | <input type="checkbox"/> simple filter |
| <input type="checkbox"/> rename nodes | <input type="checkbox"/> export image | <input type="checkbox"/> screenshot image |
| <input type="checkbox"/> groupin manually | | |

3.9 Success



3.10 Notes:



4 Exercise 3

This provenance graph is an abstract representation of a report based on twitter data.



NONE



2230979588 0007

4.1 Exercise 3 general notes:

4.2 Describe the graph to me

- ☐ multiple twitter feeds
- ☐ three reports
- ☐ twitter feeds have data from different days
- ☐ advice-report created using data from three reports
- ☐ data from two users

4.3 Success

☐ ☐ ☐ ☐ ☐

4.4 Notes:

4.5 You want to modify the graph in order to convey the following information to a non-technical user: An advice report was created using data from 3 other reports based on twitter data from two different users.

- ☐ rename nodes
- ☐ grouping Y data
- ☐ groupin manually
- ☐ export image
- ☐ grouping twitter feeds
- ☐ grouping X data
- ☐ silmple filter

4.6 Success

☐ ☐ ☐ ☐ ☐

NONE



2230979588 0008

4.7 Notes:

4.8 You want to show to user X that their data was used in the advice report by sharing an image of the provenance with them. Modify the graph to best represent this information.

- | | | |
|---|---------------------------------------|--|
| <input type="checkbox"/> rename nodes | <input type="checkbox"/> export image | <input type="checkbox"/> simple filter |
| <input type="checkbox"/> groupin manually | <input type="checkbox"/> group y data | <input type="checkbox"/> hide report 3 |

4.9 Success

☐ ☐ ☐ ☐ ☐

4.10 Notes:



NONE



2230979588 0009

5 SUS Questionnaire

Answer the questions below on a scale of 1: Strongly disagree to 5: Strongly agree. This survey is more effective if you record immediate response to each item rather than thinking about them for an extended period.

5.1

- | | | | | | | | |
|---|---|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|---|
| I think that I would like to use this system frequently | 1 | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | 5 |
| I found the system unnecessarily complex | 1 | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | 5 |
| I thought the system was easy to use | 1 | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | 5 |
| I think that I would need the support of a technical person to be able to use this system | 1 | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | 5 |
| I found the various functions in this system were well integrated | 1 | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | 5 |
| I thought there was too much inconsistency in this system | 1 | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | 5 |
| I would imagine that most people would learn to use this system very quickly | 1 | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | 5 |
| I found the system very cumbersome to use | 1 | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | 5 |
| I felt very confident using the system | 1 | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | 5 |
| I needed to learn a lot of things before I could get going with this system | 1 | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | 5 |



NONE

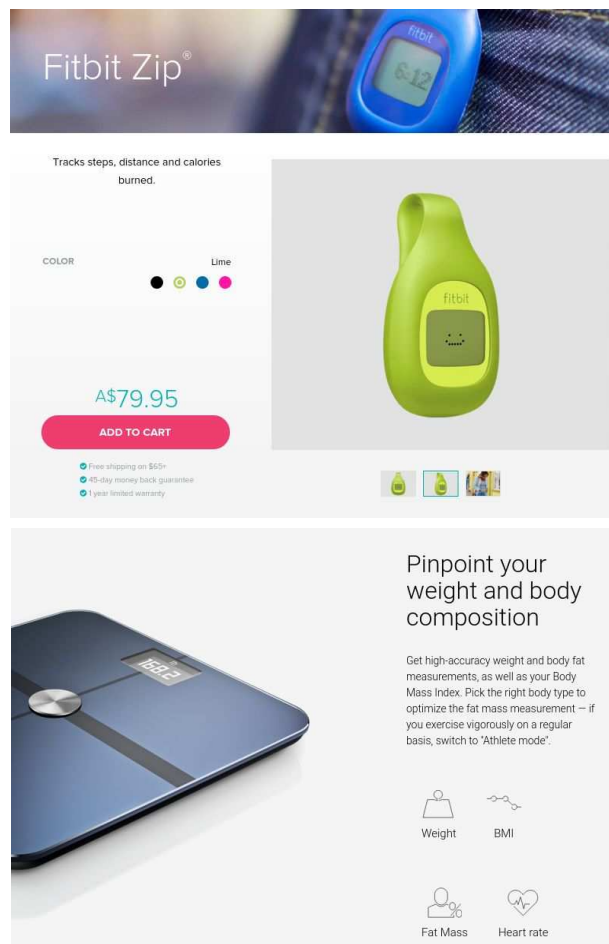


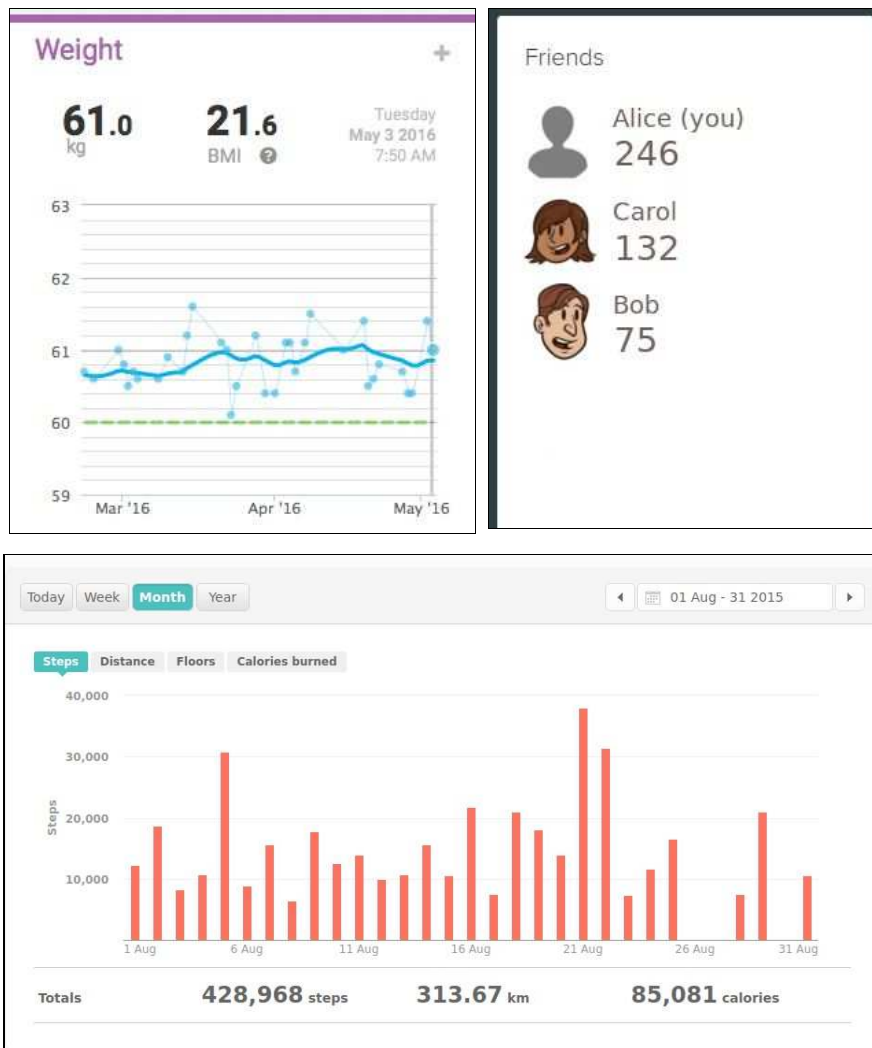
2230979588 0010

Appendix C. Participant Information Sheet

Exercise 1 Infosheet

Suppose you are Alice. You have been using your FitBit and Withings scales for 3 years to reduce your weight to 61kg and to increase your physical activity to 8000 steps a day. You have just linked your FitBit account to two friends, Bob and Carol. Recently FitBit introduced a fitness score that is shared and ranked with your friends... This makes you wonder just how your score has been calculated.





Clockwise from top are different aspects of your fit-bit dashboard: Weight information generated from your Withings scale. Your fitness score compared to your friends. Step generated pulled from you fitbit.

Using the provenance graph [here](#). Answer the following questions:

1. What data about you is used to calculate your score?
2. Identify what aspects of the provenance graph map to your fitness dashboard.
3. What processes does your raw step data go through before been used to calculate the fitbit_score?
4. How do you expect your raw fitbit data would be represented once it got to score_generator?
5. What processes does your Withings scale data go through before been used to calculate the fitbit_score?
6. How do you expect your weight data would be represented once it got to score_generator?
7. Are you concerned about how you data is used to calculate the fitbit score. Explain your reasoning.

Exercise 2 Infosheet

You are a researcher creating a citizen-data-report that recommends strategies for improving the fitness of your local community. You have step, location, weight and calorie information about members of the community. You are using this information to create multiple reports regarding fitness and weight information which will be used to support a final report addressed to the community on strategies that can be used to improve overall fitness.

Using the provenance of the final strategy report [here](#). Answer the following questions.

1. Describe the graph to me
2. You want to share how the report was generated with a colleague however you want to hide information about the cohort for privacy. Modify the graph in order to hide identifying information about participants. Then save an image of it. Note: try using the filter function for grouping multiple nodes
3. You want to show Alice how her information is been used by sharing an image of the provenance with her that illustrates the lineage of the citizen report and how her data influences it. You don't want Alice to see details about other people in the cohort.

Exercise 3 Infosheet

This provenance graph is an abstract representation of a report based on twitter data.

1. Describe the graph to me
2. You want to modify the graph in order to convey the following information to a non-technical user: An advice report was created using data from 3 other reports based on twitter data from two different users.
3. You want to show to user X that their data was used in the advice report by sharing an image of the provenance with them. Modify the graph to best represent this information.

Appendix D. PROV File: Alice's Fitness Score

```
1 document
2
3 agent(ir:Alice, [fullname="Alice Andrews"])
4
5 entity(ir:fitness_score)
6 entity(ir:weight_success)
7 entity(ir:aggregated_weight_data)
8 entity(ir:weight_goal)
9 entity(ir:weight_scale_data)
10 entity(ir:weight_manual_data)
11 entity(ir:step_achievement)
12 entity(ir:step_data)
13 entity(ir:step_goal)
14 entity(ir:step_tracker_data)
15
16 activity(ir:score_generator)
17 activity(ir:weight_analyser)
18 activity(ir:weight_aggregator)
19 activity(ir:step_analyser)
20 activity(ir:step_transformer)
21
22 wasGeneratedBy(ir:fitness_score, ir:score_generator)
23 used(ir:score_generator, ir:weight_success)
24 wasGeneratedBy(ir:weight_success, ir:weight_analyser)
25 used(ir:weight_analyser, ir:aggregated_weight_data)
26 used(ir:weight_analyser, ir:weight_goal)
27 wasGeneratedBy(ir:aggregated_weight_data, ir:weight_aggregator)
28 used(ir:weight_aggregator, ir:weight_scale_data)
29 used(ir:weight_aggregator, ir:weight_manual_data)
30
31 used(ir:score_generator, ir:step_achievement)
32 wasGeneratedBy(ir:step_achievement, ir:step_analyser)
33 used(ir:step_analyser, ir:step_data)
34 used(ir:step_analyser, ir:step_goal)
35 wasGeneratedBy(ir:step_data, ir:step_transformer)
36 used(ir:step_transformer, ir:step_tracker_data)
37
38 wasAttributedTo(ir:weight_goal, ir:Alice)
39 wasAttributedTo(ir:weight_manual_data, ir:Alice)
40 wasAttributedTo(ir:step_goal, ir:Alice)
41 wasAttributedTo(ir:weight_scale_data, ir:Alice)
42 wasAttributedTo(ir:step_tracker_data, ir:Alice)
43
44 endDocument
```

Appendix E. PROV File: IR-Baseline

```

1 document
2
3 prefix xsd <http://www.w3.org/2001/XMLSchema>
4 prefix prim <http://openprovenance.org/primitives#>
5 prefix prov <http://www.w3.org/ns/prov#>
6 prefix xsi <http://www.w3.org/2001/XMLSchema-instance>
7 prefix pc1 <http://www.ipaw.info/pc1/>
8
9 prefix ir <http://www.ir.ncl.ac.uk/>
10
11 //
12 // entities
13 //
14
15 // Twitter Feed
16 entity(ir:TwitterFeed-time-1, [from="2012-09-27T09:00:00",
17   to="2012-10-27T09:30:00", Status="Unclassified"])
18 entity(ir:TwitterFeed-time-2, [from="2012-12-28T09:00:00",
19   to="2013-01-28T09:30:00", Status="Unclassified"])
20 entity(ir:TwitterFeed-time-3, [from="2013-02-29T09:00:00",
21   to="2013-03-29T09:30:00", Status="Unclassified"])
22
23 // X data
24 entity(ir:X-tweets-1)
25 entity(ir:X-tweets-2)
26 entity(ir:X-tweets-3)
27
28 // Y: data
29 entity(ir:Y-tweets-1)
30 entity(ir:Y-tweets-2)
31 entity(ir:Y-tweets-3)
32
33 // classified information
34
35 entity(ir:X-profile)
36 entity(ir:Y-profile)
37
38 //entity(ir:merging-algorithm, [prov:type="plan"])
39
40 // summaries
41
42 entity(ir:X-summary)
43 entity(ir:Y-summary)
44
45 // intermediate merged results
46
47 entity(ir:report-1)
48 entity(ir:report-2)
49 entity(ir:report-3)
50
51 // final results

```

```
50
51 entity(ir:advice-report)
52
53 //
54 // activities
55 //
56
57 // X: extraction
58
59 activity(ir:query-X-Time-1)
60 activity(ir:query-X-Time-2)
61 activity(ir:query-X-Time-3)
62
63 // Y: extraction
64 activity(ir:query-Y-Time-1)
65 activity(ir:query-Y-Time-2)
66 activity(ir:query-Y-Time-3)
67
68 //wasAssociatedWith(ir:query-X-Time-1, twitterMonitor,-)
69 //wasAssociatedWith(ir:query-X-Time-2, twitterMonitor,-)
70 //wasAssociatedWith(ir:query-X-Time-3, twitterMonitor,-)
71 //wasAssociatedWith(ir:query-Y-Time-1, twitterMonitor,-)
72 //wasAssociatedWith(ir:query-Y-Time-2, twitterMonitor,-)
73 //wasAssociatedWith(ir:query-Y-Time-3, twitterMonitor,-)
74
75
76 // merging
77
78 activity(ir:consolidate-X)
79 activity(ir:consolidate-Y)
80 activity(ir:report-editing)
81
82 // analytics activity
83
84 activity(ir:analytics1)
85 activity(ir:analytics2)
86 activity(ir:analytics3)
87
88
89 //
90 // agents
91 //
92 //agent(ir:Alice)
93 //agent(ir:Bob)
94 //agent(ir:Charlie)
95 //agent(ir:David)
96
97 //agent(ir:K)
98 //agent(ir:L)
99 //agent(ir:M)
100
101 //agent(ir:twitterMonitor)
102
103
```

```

104 //actedOnBehalfOf(ir:Alice, ir:David, ir:analytics1)
105 //actedOnBehalfOf(ir:Bob, ir:David, ir:analytics2)
106 //actedOnBehalfOf(ir:Charlie, ir:David, ir:analytics3)
107
108 // attributions and associations, delegation
109 //wasAssociatedWith(ir:report-editing, ir:K,-)
110 //actedOnBehalfOf(ir:K, ir:L, ir:report-editing)
111
112 //actedOnBehalfOf(ir:Alice, ir:M)
113 //actedOnBehalfOf(ir:Bob, ir:M)
114 //actedOnBehalfOf(ir:Charlie, ir:M)
115 //actedOnBehalfOf(ir:David, ir:M)
116 //actedOnBehalfOf(ir:K, ir:M)
117 //actedOnBehalfOf(ir:L, ir:M)
118
119 //wasAssociatedWith(ir:consolidate-X, ir:Charlie,
    ir:merging-algorithm)
120 //wasAssociatedWith(ir:consolidate-Y, ir:Charlie,
    ir:merging-algorithm)
121
122 //wasAssociatedWith(ir:analytics1, ir:Alice,-)
123 //wasAssociatedWith(ir:analytics2, ir:Bob,-)
124 //wasAssociatedWith(ir:analytics3, ir:Charlie,-)
125
126
127 // initial extraction activity
128
129 used(ir:query-X-Time-1,ir:TwitterFeed-time-1,-)
130 used(ir:query-X-Time-2,ir:TwitterFeed-time-2,-)
131 used(ir:query-X-Time-3,ir:TwitterFeed-time-3,-)
132 used(ir:query-Y-Time-1,ir:TwitterFeed-time-1,-)
133 used(ir:query-Y-Time-2,ir:TwitterFeed-time-2,-)
134 used(ir:query-Y-Time-3,ir:TwitterFeed-time-3,-)
135
136 // initial extraction results
137
138 wasGeneratedBy(ir:X-tweets-1,ir:query-X-Time-1,2013-06-18T11:10:00)
139 wasGeneratedBy(ir:X-tweets-2,ir:query-X-Time-2,2013-06-18T11:10:00)
140 wasGeneratedBy(ir:X-tweets-3,ir:query-X-Time-3,2013-06-18T11:10:00)
141 wasGeneratedBy(ir:Y-tweets-1,ir:query-Y-Time-1,2013-06-18T11:10:00)
142 wasGeneratedBy(ir:Y-tweets-2,ir:query-Y-Time-2,2013-06-18T11:10:00)
143 wasGeneratedBy(ir:Y-tweets-3,ir:query-Y-Time-3,2013-06-18T11:10:00)
144
145 // merging
146
147 used(ir:consolidate-X,ir:X-tweets-1,-)
148 used(ir:consolidate-X,ir:X-tweets-2,-)
149 used(ir:consolidate-X,ir:X-tweets-3,-)
150 used(ir:consolidate-X,ir:X-profile,-)
151
152 used(ir:consolidate-Y,ir:Y-tweets-1,-)
153 used(ir:consolidate-Y,ir:Y-tweets-2,-)
154 used(ir:consolidate-Y,ir:Y-tweets-3,-)
155 used(ir:consolidate-Y,ir:Y-profile,-)

```

```
156
157
158 // merge generates
159
160 wasGeneratedBy(ir:X-summary,ir:consolidate-X,2013-06-18T11:10:00)
161 wasGeneratedBy(ir:Y-summary,ir:consolidate-Y,2013-06-18T11:10:00)
162
163 // analytics uses
164
165 used(ir:analytics1,ir:X-summary,-)
166 used(ir:analytics1,ir:Y-summary,-)
167 used(ir:analytics2,ir:X-summary,-)
168 used(ir:analytics2,ir:Y-summary,-)
169 used(ir:analytics3,ir:X-summary,-)
170 used(ir:analytics3,ir:Y-summary,-)
171
172 // analytics generates
173
174 wasGeneratedBy(ir:report-1,ir:analytics1,2013-06-18T11:10:00)
175 wasGeneratedBy(ir:report-2,ir:analytics2,2013-06-18T11:10:00)
176 wasGeneratedBy(ir:report-3,ir:analytics3,2013-06-18T11:10:00)
177
178 // final merging
179
180 used(ir:report-editing,ir:report-1,-)
181 used(ir:report-editing,ir:report-2,-)
182 used(ir:report-editing,ir:report-3,-)
183
184 // generating the final result
185
186 wasGeneratedBy(ir:advice-report, ir:report-editing,
187               2013-06-18T11:10:00)
188
189 endDocument
```

Appendix F. Concrete codes

TABLE 1. List of question codes and descriptions of each.

Code	Description
grouping	Activities that involved users grouping or ungrouping nodes (by any means)
grouping-group	When a user grouped a node
grouping-ungroup	When a user ungrouped a node
grouping-complex	When a user grouped a complex set of nodes
graph	Activities related to understanding a graph conceptually
graph-explain	Notice and comment on information in the graph. An example is in exercise one where some users noticed that weight data is both manually and automatically logged. This information is not usually relevant to the question.
graph-understanding	Correctly understand a concept from the graph, usually directly in response to a question.
lineage	Correctly identifying information related to the lineage of a node.
lineage-identify	Correctly identified simple lineage.
lineage-implied	Correctly identified lineage passing through multiple entities.
lineage-complex	Correctly identified lineage that was complex, usually identifying multiple nodes.
interface	When a user uses a specific bit of the ProvOwl interface.
interface-export-image	Using the export -> image menu item to save an image of a graph.
interface-os-screenshot	Using the operating system screenshot functionality to save an image of a graph.
interface-rename	Using the rename function (found in the contextual links in a nodes details) to rename a node.
interface-manual-group	Manually group nodes by ctrl+clicking multiple nodes and selecting <i>group nodes</i> from the contextual menu.
interface-search-group	Grouping nodes by searching for the nodes they want to use using the search function and then grouping them.
privacy	Whether a user was privacy conscious at all in relation to personal data.

TABLE 2. Scenario one subtasks

Exercise	Question/activity	Code
1	What data about you is used to calculate your score?	
1.1	weight_scale_data	lineage-identify
1.2	weight_manual_data	lineage-identify
1.3	weight_goal	lineage-identify
1.4	step_tracker_data	lineage-identify
1.5	step_goal	lineage-identify
1.6	question Alice was attributed to fitbit_score	lineage-implied
1.7	notice manual and automatic logging was used	explain
1.8	ungrouped nodes	graph-explain
2	Identify what aspects of the provenance graph map to your fitness dashboard.	
2.1	raw_scale	graph-understanding
2.2	raw_steps	graph-understanding
2.3	fitness_score	graph-understanding
2.4	ungrouped nodes	grouping-ungroup
3	What processes does your raw step data go through before been used to calculate the fitbit_score?	
3.1	step_transformer	lineage-identify
3.2	step_data	lineage-identify
3.4	step_analyser	lineage-complex
3.5	step_achievement	lineage-identify
3.6	score_generator	lineage-identify
3.7	fitness_score	lineage-identify
3.8	ungrouped nodes	grouping-ungroup
4	How do you expect your raw fitbit data would be represented once it got to score_generator?	
4.1	understands data transformation	graph-understanding
4.2	not at all	graph-understanding
4.3	in anonymised form	graph-understanding

4.4	ungrouped nodes	grouping-ungroup
5	What processes does your Withings scale data go through before been used to calculate the fitbit_score?	
5.1	aggregated	lineage-identify
5.2	analysed	lineage-identify
5.3	algorithm to determine success	lineage-identify
5.4	ungrouped nodes	grouping-ungroup
5.5	identified manual and automatic input	graph-explain
6	How do you expect your weight data would be represented once it got to score_generator?	
6.1	understands data transformation	graph-understanding
6.2	not at all	graph-understanding
6.3	in anonymised form	graph-understanding
6.4	ungrouped nodes	grouping-ungroup
7	Are you concerned about how you data is used to calculate the fitbit score. Explain your reasoning.	
7.1	Worried about weighting	graph-explain
7.2	Worried about privacy	privacy
7.3	Not-concerned	privacy

TABLE 3. Scenario two subtasks

Exercise	Question/activity	Code
1	Describe the graph to me	
	people with identical data	lineage-identify
	cohort data anonymised	lineage-identify
	averages created from anonymised cohort	lineage-identify
	two reports created	lineage-identify
	citizen-science-report based on two reports	lineage-identify
	grouping with simple filter	interface-search-group
	grouping manually	interface-manual-group

2	You want to share how the report was generated with a colleague however you want to hide information about the cohort for privacy. Modify the graph in order to hide identifying information about participants.	group
	Group nodes ungroup nodes simple filter rename nodes export image screenshot image grouping manually	grouping-group grouping-ungroup interface-search-group interface-rename interface-export-image interface-os-screenshot interface-manual-group
3	You want to show Alice how her information is been used by sharing an image of the provenance with her that illustrates the lineage of the citizen report and how her data influences it. You do not want Alice to see details about other people in the cohort.	
	Group nodes ungroup nodes simple filter rename nodes export image screenshot image grouping with simple filter grouping manually	grouping-group grouping-ungroup interface-search-group interface-rename interface-export-image interface-os-screenshot interface-search-group interface-manual-group

TABLE 4. Scenario three subtasks

Exercise	Question/activity	Code
1	Describe the graph to me	
	multiple twitter feeds twitter feeds have data from different days data from two users	lineage-identify interface-details lineage-identify

	three reports advice-report created using data from three reports	lineage-identify lineage-identify
2	You want to modify the graph in order to convey the following information to a non-technical user: An advice report was created using data from 3 other reports based on twitter data from two different users.	
	rename nodes export image grouping X data grouping Y data grouping twitter feeds grouping with simple filter grouping manually	interface-rename interface-export-image grouping-group grouping-group grouping-complex interface-search-group interface-manual-group
3	You want to show to user X that their data was used in the advice report by sharing an image of the provenance with them. Modify the graph to best represent this information.	
	rename nodes export image grouping with simple filter grouping manually group y data hide report 3	interface-rename interface-export-image interface-search-group interface-manual-group grouping-complex grouping-complex