UNIVERSITY OF SOUTHAMPTON

FACULTY OF PHYSICAL SCIENCES AND ENGINEERING

Electronics and Computer Science

The Domain Agnostic Generation of Natural Language Explanations from Provenance Graphs

by

Darren Paul Richardson

Thesis for the degree of Doctor of Philosophy

June 2018

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF PHYSICAL SCIENCES AND ENGINEERING
Electronics and Computer Science

Doctor of Philosophy

THE DOMAIN AGNOSTIC GENERATION OF NATURAL LANGUAGE EXPLANATIONS FROM PROVENANCE GRAPHS

by Darren Paul Richardson

In a data-driven world, being able to record from where data was derived, and by whom is key. The way to represent this information, provenance, on the Web has been standardised by the World Wide Web Consortium as PROV. Furthermore, once provenance has been recorded, it is often necessary to be able to present it back to users. In the state-of-the-art, the interfaces to such provenance tend to be diagrammatic, or rely on very application-specific template-based natural language generation. Both of these approaches have their drawbacks, motivating the search for techniques for generating natural language explanations from domain-generic provenance graphs. This work presents several contributions to the state-of-the-art in this regard. Firstly it presents a novel template-based architecture for natural language generation. This is followed by the novel application of set-cover optimisation techniques to the challenge of sentence selection. Thirdly, this work extends previous research into the role of URIs for lexicalising Linked Data resources, making use of the specific nature of PROV instance data to inform the heuristics used. Fourthly, these techniques are then evaluated in a user study demonstrating that they improve upon the state-of-the-art across the three dimensions of grammatical correctness, fluency, and comprehensibility. This evaluation also showed that the participants preferred the sentences generated using these techniques 56.4% of the time. Following on from these advances, an investigation is conducted into how to structure larger natural language explanations of provenance graphs. This is done by inviting a number of provenance experts to describe a sequence of provenance graphs presented diagrammatically, and analysing the way they approach this task. This reveals that the responses of the experts correlated strongly with the visual layout of the diagrams, and also that the experts were split as to whether to structure those explanations in a chronological or anti-chronological order. Finally, a further study was conducted to investigate how chronology affects the perceived quality of the generated natural language explanations, revealing that in aggregate the participants considered the chronological ordering to be more logical. This dissertation concludes with a summary of the contributions made to the state-of-the-art, as well as by proposing a number of possible areas for future research.

Contents

List of Figures

List of Tables

Declaration of Authorship

I, Darren Paul Richardson , declare that the thesis entitled *The Domain Agnostic Generation of Natural Language Explanations from Provenance Graphs* and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- parts of this work have been published as: Richardson et al. (2014), Richardson et al. (2015), and Richardson and Moreau (2016).

Signed:.....

Date:.....

Acknowledgements

Many people have contributed in multifarious ways to the success of this research, and in supporting me whilst I pursued it. First and foremost, I would like to thank my supervisors, Luc Moreau, Paul Smart, and Nigel Shadbolt, who have all contributed their time, energy, and individual passion for their subjects to the success of this project.

I would also like to acknowledge the help and advice given by colleagues from the International Technology Alliance, and in particular David Mott and Dave Braines from IBM UK and Cheryl Giammanco from the US Army Research Laboratory who have all lent their valuable experience.

Of particular value to me during my PhD has been the friendship from within the Web and Internet Science research group at the University of Southampton, but in particular the 3:30pm coffee crew, who have been at times a sounding board for ideas, and at other times have lent an understanding ear to my woes, all the while being supportive and unendingly interesting people: Dave Newman, Tyler Ward, Phil Basford, Graeme Bragg, Tom Blount, Rikki Prince, Charlie Hargood, Huw Fryer, Ash Smith, Reena Pau, Chris Gutteridge, Jonathon Hare, Tim Chown, and Kirk Martinez.

In addition, I would like to thank all those at Life Church Southampton who have done a fantastic job of keeping my spirits up during the hardest parts of my PhD, and to those at King's Church Horsham who have supported me during the process of writing this thesis. You have all been a true blessing.

Finally, I would like to thank my parents, who have always been supportive of this effort, even when they've not completely understood it. I am particularly grateful for their patience in the last few months of writing, when I have been completely unsociable and have thoroughly annexed their dining room.

To know wisdom and instruction,
to understand words of insight,
to receive instruction in wise dealing,
in righteousness, justice, and equity;
to give prudence to the simple,
knowledge and discretion to the youth —
Let the wise hear and increase in learning,
and the one who understands obtain guidance,
to understand a proverb and a saying,
the words of the wise and their riddles.
The fear of the LORD is the beginning of knowledge;
fools despise wisdom and instruction.

Proverbs 1:2-7 (English Standard Version)

Nomenclature

API	Application Programming Interface
AVM	Attribute-value Matrix
IETF	Internet Engineering Task Force
IRI	Internationalised Resource Identifier
JSON	JavaScript Object Notation
LD	Linked Data
NL	Natural Language
NLG	Natural Language Generation
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
OPM	Open Provenance Model
OWL	Web Ontology Language
POS	Part-of-speech
PROV	W3C Provenance Framework
RDF	Resource Description Framework
REST	Representational State Transfer
RST	Rhetorical Structure Theory
SPARQL	SPARQL Protocol and RDF Query Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
UUID	Universally Unique Identifier
W3C	World Wide Web Consortium
XML	Extensible Markup Language

Chapter 1

Introduction

Data is not intrinsically valuable. First, in the same way crude oil needs to be filtered, fractioned, refined, tanked, and transported before it can be used as fuel in a car, data is only useful once it has been interpreted and given meaning. The field of information technology is driven by a desire to be able to process data in greater volumes, more quickly, at lower costs, and with a greater degree of insight. The corresponding advances in technology have, in turn, resulted in benefits for businesses, researchers, and governments, as well as individuals and whole societies.

Businesses are ever wanting to increase revenues whilst reducing overheads and costs-per-sale. Conventional wisdom states that by gaining a better understanding of their customers, internal processes, and competitors, businesses will be able to do just that. IBM Software (2014) claims that retailers using the information provided by "Big Data" analytics could increase their operating margins by more than 60%. Beyond the hype, there are other reasons why companies would want, or need, to store data. Regulatory compliance often requires businesses to keep records, and there can be legal ramifications if these are not adequately kept — an example is the food industry, where governments are increasingly introducing regulations to ensure the traceability of the human food chain (Trienekens and Zuurbier, 2008).

Furthermore, there are a number of factors that motivate governments to release data. Firstly, in democratic society, releasing data about how a government operates and makes decisions increases accountability. Correspondingly, many countries' legislatures have passed freedom of information laws, mandating that governments release information in response to requests and, in the UK at least, giving them a duty to proactively seek to publish such data (Mendel, 2008). Secondly, governments have the resources and regulatory levers necessary to collect much larger and more comprehensive datasets than researchers or most businesses, and by releasing this data they increase its utility and value to the tax-payer.

The United Kingdom government, for example, has committed to openly publishing data where it deems appropriate with four key goals (UK Cabinet Office, 2013): government accountability and the ability to assess the efficacy of government policy; reuse within government leading to increased efficiency; reuse external to government leading to increased economic growth; and use by members of the public to help them make appropriate choices with respect to government services, such as the National Health Service.

The case for individuals needing to handle large quantities of data is somewhat more limited. However, the increasing instrumentation of our lives through devices such as internet-connected smartphones, pedometers, heart-rate monitors, and sleep trackers, as well as of our homes through a whole range of devices from intelligent thermostats to Smart TVs, means that individuals are collecting and storing increasingly large amounts of data — though still relatively small compared to the data collected by businesses and governments. They are collecting this data because it has some perceived benefit to them, from losing weight to reducing energy bills, but it also has massive potential to businesses and researchers when collated with similarly structured datasets. This is, in fact, part of the business model for companies such as Fitbit[1], who are able to sell the aggregated and anonymised activity data from their customers to interested third parties.

With so much data being generated, processed, and shared, it is becoming increasingly important to track how a piece of information was derived, what has been done to it since its creation, and who or where it came from — *data provenance*. The reasons for wanting to track this range from being able to prove regulatory compliance (Hasan et al., 2007), to giving scientific or even monetary credit where due (Dragan et al., 2014), to assuring the quality of a piece of data (Hartig and Zhao, 2009), amongst others.

Meanwhile, researchers have long understood the importance of provenance and workflows in increasing the value of their scientific output (De Roure et al., 2007), with the editor-in-chief of *Science* recently noting in McNutt (2015) that while scientists might have different interpretations of data, as long as the data is published alongside the researcher's methodology and analysis, there is an opportunity for critical, independent critique and reproducibility. As an example, the 2014 National Climate Assessment[2] provided provenance for all its data sources, helping lend credibility to its findings in a politically charged and contentious environment. Additionally, large datasets are expensive to initially collect, curate, and if necessary keep up-to-date. Improvements in information technology help reduce this barrier to research, and potentially even allow researchers to monetise their databases by licensing the data to other interested parties, funding further research. For this reuse to occur, however, the quality of the data needs

[1] `http://fitbit.com` — See also their privacy policy, as of 13th March 2015: `https://web.archive.org/web/20150313095700/http://www.fitbit.com/privacy`

[2] `http://nca2014.globalchange.gov`, with the supporting datasets including provenance available at `https://data.globalchange.gov`

to be assessable, and reviewing provenance annotations is potentially one part of that process.

Finally, the specific motivation for this work comes from the military domain. This work was funded by the International Technology Alliance in Network and Information Sciences — a collaboration funded by the United Kingdom's Defence Science and Technology Laboratory, and the United States' Army Research Laboratory. Having had a number of opportunities to talk with military commanders in an informal setting, it became apparent that they wanted a better way of knowing where the information they were handling came from. On its journey from sensor to decision-maker, a piece of information goes through multiple rounds of analysis and summarisation. Each of these steps helps obscure the original source of the information, and even has the possibility of subtly changing its meaning. Commanders making life and death decisions want to know that they can trust the information upon which they are basing those decisions. By the time this information makes its way to decision-makers, it is typically in the form of a report, either written or presented verbally, perhaps over a communications channel. Consequently, were a way developed to automatically incorporate natural language provenance traces into these reports, in a way that the commander could clearly understand, this would potentially be of great benefit — allowing commanders to have a more appropriate level of confidence in the reports they read, and a greater degree of confidence in the decisions that they make.

1.1 Introduction to the Semantic Web and PROV

The Semantic Web was proposed by Berners-Lee (1999) as a vision of what he wanted the World Wide Web to develop into. At the time, much of the Web was made up of static documents, and whilst they contained a great deal of valuable information, it was computationally expensive for any other system to be able to make use of that information in any sort of automated fashion. This problem has been made easier since then, since many many Websites now offer custom APIs, allowing developers to interact with them programmatically. However, these usually require a developer to write custom code that integrates with these custom APIs. This, also, has been made easier with the adoption of common API design patterns, like REST (Fielding, 2000), but it is nevertheless still not as simple to utilise the data from another system as it might be. The aim of Berners-Lee's vision is the development of a Web where it would be possible for a system to reuse data from all over the Web, without needing to write these custom integrations for every data source.

The way he proposed achieving this is now called Linked Data, and provides a common framework — the Resource Description Framework, described in Chapter 2 — as common ground for systems to transfer information about anything, without the need for

complicated custom integrations. However, by allowing information to be interchanged easily between systems, and by expecting that a highly interdependent network of data will become the norm, questions have arisen about how one can assess the reliability and trustworthiness of any of this data, or anything derived from it. Provenance has been proposed as one possible solution.

Provenance, and in particular provenance on the Web, is introduced in greater detail in Chapter 2. For now, it suffices to say that provenance can be defined as "a record that describes the people, institutions, and activities involved in producing, influencing, or delivering a piece of data or a thing." (Moreau and Missier, 2013a). In addition, the World Wide Web Consortium (W3C) has released a suite of recommendations standardising the representation of provenance on the Web, called PROV, which is the focus of this work. With PROV, provenance is represented as a mathematical graph, with edges representing the provenance relationships between the nodes representing resources. These edges and nodes are denoted by URIs, meaning PROV provenance is a form of Linked Data (described in Chapter 2).

PROV defines three core concepts — *entities*, *activities*, and *agents*. The following three definitions are quoted verbatim from the W3C PROV-DM recommendation (Moreau and Missier, 2013a):

Entity An entity is a physical, digital, conceptual, or other kind of thing with some fixed aspects; entities may be real or imaginary.

Activity An activity is something that occurs over a period of time and acts upon or with entities; it may include consuming, processing, transforming, modifying, relocating, using, or generating entities.

Agent An agent is something that bears some form of responsibility for an activity taking place, for the existence of an entity, or for another agent's activity.

As might be suggested by the above definitions, PROV also specifies a number of relationships between those concepts. For example, activities can *use* entities. Figure 1.1 shows a subset of these relationships, as well as demonstrating which of the three core types they relate to one another[3].

Together, the suite of W3C PROV recommendations is designed to allow for the representation and interchange of provenance data on the Web, and builds upon other Semantic Web technologies described in Chapter 2. That is to say, the PROV recommendations are primarily designed to facilitate the communication of data provenance from machine to machine. In addition, however, one of the PROV recommendations

[3]Figure 1.1 is taken from PROV-PRIMER (http://www.w3.org/TR/prov-primer/): Copyright © 2013 W3C® (MIT, ERCIM, Keio, Beihang), All Rights Reserved. W3C liability, trademark and document use rules apply.

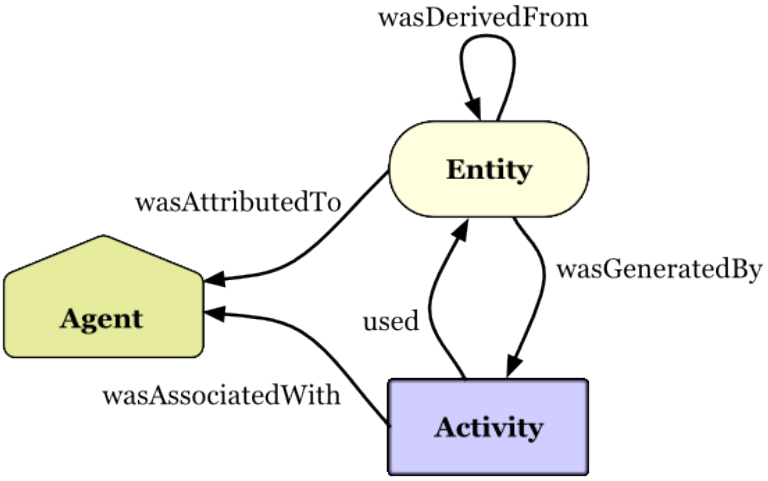


Figure 1.1: The core concepts and relations of the PROV Data Model.

defines a syntax called PROV-N (Moreau and Missier, 2013b), which describes itself as "aimed at human consumption". Nevertheless, PROV-N was not intended to be used by casual users of a system, requiring an understanding of the PROV model, as well as other Semantic Web technologies.

1.2 Problem statement

As computational systems are beginning to use PROV more intensively, a need is arising to be able to communicate that data to users in a manner they can understand. Diagrammatic representations are often excluded by the need to be able to communicate with all users, including those with no training, experience, or formal knowledge of provenance; diagrammatic representations, such as the relatively simple Figure 1.2, contain elements and features that can only be understood by somebody with a reasonably advanced understanding of provenance, as the meanings of colours and shapes are non-obvious — the yellow ellipses represent *entities*, the blue rectangles represent *activities*, and the orange pentagons represent *agents*, as described in Section 1.1.

On the other hand, all able-minded adult users can be expected to be able to communicate using language, and whereas they cannot be expected to be able to comprehend provenance communicated diagrammatically particularly easily, natural language contains all the features needed to adequately describe provenance in a way that can be understood by individuals across the population.

In addition, there are a number of scenarios in which linguistic communication simply makes more sense than a diagrammatic approach. For example, one should not be expected to have to spend too much time looking at a screen whilst driving, and in some cases this might even be considered illegal. Additionally, within the military context

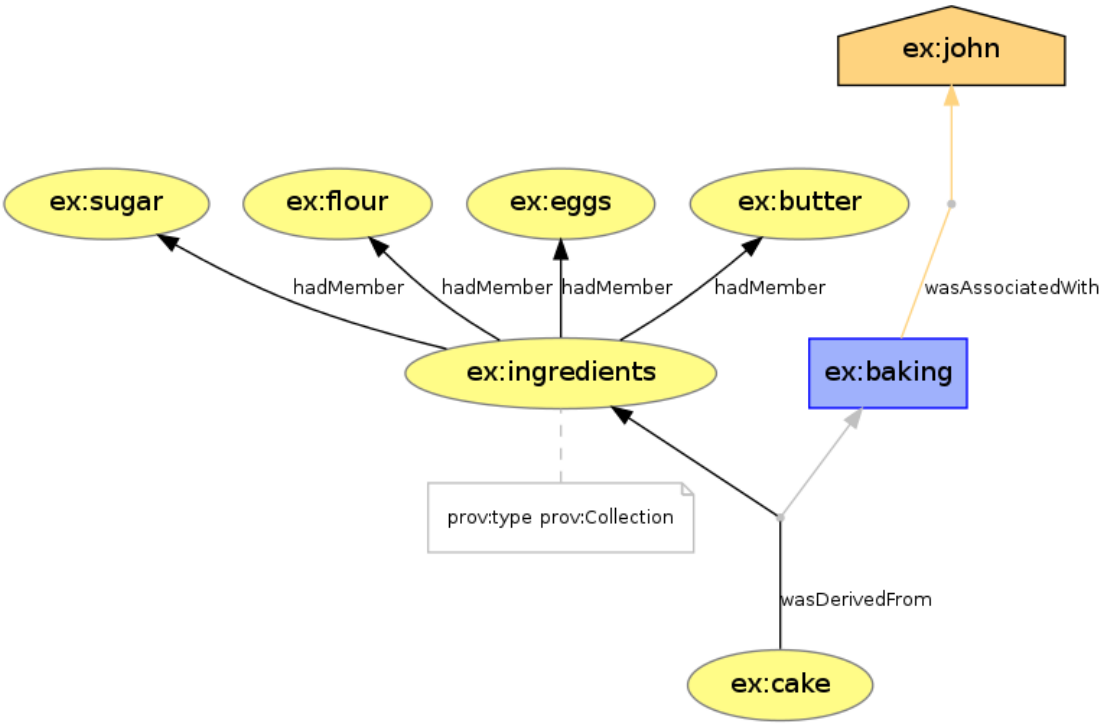


Figure 1.2: An example PROV graph, in a common diagrammatic representation format. It shows the process of a cake being baked by an agent called John, from a collection of ingredients.

described earlier, many pieces of information can be passed onto decision-makers over an audio-only communications channel, necessitating the use of a language-based approach.

Consequently, the research problem of this work is to develop an approach, and associated techniques, for generating natural language explanations from provenance graphs using the PROV data model (as introduced in Section 1.1 and described in more detail in Chapter 2). For the purposes of this work, however, this problem is only addressed with respect to generating natural language explanations from provenance in English, though many of the techniques and principles involved could potentially generalise to other languages.

In order to generate a linguistic explanation, the abstract concepts in a provenance graph first need to be mapped to lexemes in the target language. This means that the largest research problem is to find an appropriate technique for extracting the available linguistic information from the provenance graph, because these graphs do not necessarily formally encode any linguistic information. This lexicalisation challenge is explored in greater detail in Section 2.3, and there are a number of different ways one might approach it, such as mandating the use of a linguistic annotation ontology. However, the need to be able to communicate PROV without also mandating the use of additional annotations or mark-up, constricts the number of options available. This work focusses on the feasibility of using the URIs denoting the PROV resources themselves as a source for this lexical

information. For example, the URI `http://dbpedia.org/resource/United_Kingdom`, contains the linguistic information needed to correctly refer to the concept it denotes — The United Kingdom — and this work proposes a number of techniques for doing this.

This is potentially feasible, because, in the same way that developers often use meaningful variable names in their code to help increase its maintainability, developers of provenance-producing systems often use meaningful URIs and URI schemas, despite not being formally required to. Clearly, an ability to exploit this information would lead to more expressive explanations, whilst, as this linguistic information is not guaranteed to be present, any approach to explanation generation needs to be able to generate meaningful, if perhaps less expressive, explanations even where this information is not present.

Furthermore, in order to ensure that the approach generalises to all PROV compliant datasets, it is necessary to restrict the features that can be used to assist the explanation generation to only those that can be reasonably expected to appear in the data produced by a PROV compliant provenance-generating system.

In summary, the requirements for the approach to provenance explanation are as follows:

1. **Linguistic explanations:** The approach needs to be able to produce explanations in natural language — specifically English — from provenance graphs that are compliant with the W3C PROV recommendations.
2. **Linguistically accurate:** The explanations produced need to be free from grammatical and orthographical errors, so as not to hinder comprehension.
3. **Accessible to casual users:** The explanations produced also need to be at a level understandable by a casual user with no deep understanding of provenance, or the PROV recommendations.
4. **Generalisable to all PROV:** The approach to producing the explanations should generalise to all PROV rather than a particular application or domain.
5. **Self-contained:** The approach must only make use of features of PROV specifically, and not rely on other schemas or ontologies, or other forms of annotating the data.

1.3 Application areas

As described in the opening to this chapter, the applications of generating explanations from provenance graphs could cover every sector of modern life, from home to business, from government to research, and from individuals to societies collectively. However,

it should be noted that natural language is not limited to textual communication, but is also an important component in audio and video communication. The approaches presented in later chapters are just as applicable for producing video or audio clips with natural language components as for generating textual explanations.

Provenance is also not restricted only to the digital world, tracking the lineage of data, but is of great importance in the physical world. Even before the recent scandal where horse-meat entered the UK food supply labelled as beef[4], researchers were identifying the importance of provenance data for tracking entities in the human food-chain (Shackell, 2008).

As previously stated, the primary motivation for this work is its potential application within the defence domain. By allowing decision-makers to understand how the information they are reading came to exist — including what assets, human or technological, were involved in its initial data capture — we potentially equip them to make better decisions in life and death scenarios. Whilst the technology presented in this dissertation is not sufficiently well developed so as to be deployable in active operational contexts, it might perhaps be the first step towards such exploitation.

1.4 Existing solutions

Whilst there is no solution at present that satisfies all of the requirements stated in Section 1.2 there are however two existing approaches to communicating provenance with human users. The first of these is to use graphical, diagrammatic representations of the provenance (such as that shown in Figure 1.2) suggested by the Provenance Working Group (2011). This has the advantage that it easily exposes most of the provenance information in a clear format. However, there are some features of PROV that are harder to represent diagrammatically, and particularly for larger provenance graphs, the diagrammatic representations can become very complex and unwieldy. Additionally, a casual user with no deep understanding of the PROV data model (described in Chapter 2) would have difficulty understanding this somewhat specialised form of representation.

Another approach is to generate textual explanations from templates (Packer and Moreau, 2015) using variable-based string-substitution. As they exist in the state-of-the-art, these templates are typically built on an application by application basis, exposing relevant parts of an application's provenance data to the user where this would be helpful. This has the advantage that the system developers are working with a much more constrained set of possibilities, and can build correspondingly more specific templates that produce clear, well formed natural language based on the information they know will be at hand. The clear downside of this approach is that it requires the development of new templates every time somebody wishes to expose some provenance. Additionally, it does

[4] `http://www.bbc.co.uk/news/uk-21335872`

not work well for systems that utilise provenance provided by an external system, where the particular content and style of the provenance is far less well-known.

1.5 Thesis statement

The current lack of capability for explaining decision making and provenance is a potential road-block in the real-world realisation of the Semantic Web vision (as described earlier in this chapter, and in more detail in Chapter 2). Despite a lack of formally encoded linguistic information, in many cases the incidental linguistic information, encoded in formally meaningless URIs, can nevertheless be used to lexicalise instances of PROV concepts.

Thesis statement: *The use of a mature natural language realisation engine, combined with linguistic information derived from data often present informally in provenance graphs, allows for richer natural language explanations being generated from these provenance graphs, as seen from the perspective of potential users.*

1.6 Research objective

The research objective is as follows:

To develop a method for generating natural language explanations from provenance data. The approach should generalise to work for all provenance information that conforms with the W3C PROV recommendations, without requiring any additional information external to those recommendations.

As described in more detail in Chapter 2, there already exist some techniques for generating natural language from Linked Data. However, these are often relatively unsuccessful, when measured against the requirements elicited in Section 1.2, because of a lack of information from which to build sentences and larger linguistic structures. Provenance, by its very nature, contains much of the information, such as temporal structure and actor roles, needed to execute this conversion more successfully, and exploring techniques for doing so is a clear area of unexplored research.

1.7 Research contributions

This work contributes to the state of the art of natural language explanations generated from provenance graphs making the following contributions:

1. A new template technique, which expands upon existing NLG theories in a manner that allows for the integration of mature off-the-shelf NLG components, is able to enrich natural language explanations generated from provenance graphs by making use of additional information that can be extracted from URIs denoting PROV resources. These templates facilitate the generation of explanations from provenance graphs in a way that is domain independent, yet producing richer language than would otherwise have been possible in the state-of-the-art.

2. A user study evaluates this approach, showing that it significantly improves the quality of the natural language explanations generated from provenance graphs in terms of grammatical correctness, fluency, and comprehensibility. Additionally, it showed that 56.4% of the time the participants reported preferring the sentences generated with this approach, as opposed to only 14.2% for the sentences generated without exploiting the informally encoded linguistic information. These higher-quality natural language explanations could potentially improve interfaces to provenance, giving users another way to understand the provenance that they are using.

3. An investigation of how experts structure their own natural language explanations from provenance graphs found that approximately half of the expert participants chose to order their responses chronologically, whilst the remainder ordered their explanations anti-chronologically.

4. A user study establishes that chronology affects the perceived quality of these natural language explanations, showing that users have a moderate preference (42.8% against 26.1%) for chronologically ordered explanations over anti-chronologically ordered explanations.

Together, these contributions to the state-of-the-art make progress towards providing an additional tool to the person or organisation wishing to make their system provenance-enabled, who will be able to offer a linguistic interface to the provenance they are storing.

In addition, the evaluations provide insight into how users perceive these explanations, helping to inform future research into linguistic provenance interfaces.

Some of these contributions are detailed in the following articles:

> Richardson, D. P. and Moreau, L. Towards the Domain Agnostic Generation of Natural Language Explanations from Provenance Graphs for Casual Users. In: 6th International Provenance and Annotation Workshop. McLean, VA, USA; 2016

Richardson, D. P., Moreau, L., and Mott, D. What's in a name? Exploiting URIs to enrich provenance explanations in plain English. In: 2015 Annual Fall Meeting of the International Technology Alliance. College Park, MD, USA; 2015

Richardson, D P, Moreau, L, and Mott, D. Beyond the Graph: Telling the Story with PROV and Controlled English. In: 2014 Annual Fall Meeting of the International Technology Alliance. Cardiff, Wales, UK; 2014.

1.8 Dissertation structure

This remainder of this dissertation is broken into five chapters:

- Chapter 2 provides the necessary background with regard to both natural language generation and provenance in order to facilitate an understanding of the contributions described in the following chapters. It begins with a description of the W3C PROV recommendations and how they are have been used in practice, before moving on to discuss existing architectures and techniques for generating natural language from structured data, as well as methodologies for assessing the performance of such systems. It concludes by considering existing approaches to lexicalising Linked Data resources.
- Meanwhile, Chapter 3 presents a novel architecture for natural language generation particularly suited for the timely generation of provenance explanations brought about through its use of templates and a novel algorithm for sentence selection.
- Chapter 4 proposes, investigates, and realises an approach to extracting the linguistic information informally encoded in URIs. This approach is then evaluated with real-world provenance data, using human participants.
- Chapter 5 investigates how human provenance experts structure explanations from provenance graphs, and following on from the results of this investigation, explores the effect of chronology on the perceived quality of generated natural language explanations.
- Finally, the report concludes with Chapter 6 summarising the core contributions made in this work and describing a number of possible avenues for further research, should the work be continued.

Chapter 2

Provenance and Explanations

The challenge of generating natural language explanations from abstract provenance data sits at the juncture of two fields within computer science: firstly, the well established field of computational linguistics, and secondly the more nascent field of Linked Data and the Semantic Web. This chapter reviews the state-of-the-art in each of these fields, insofar as they are relevant to the challenges tackled in this work. It begins in Section 2.1 with a description of Linked Data and the W3C PROV model of provenance that form the basis of this work, before going on to discuss the task of generating natural language in Section 2.2. It concludes with a discussion of the lexicalisation challenge in Section 2.3, a particular issue when generating natural language specifically from Linked Data.

2.1 The Semantic Web, Linked Data, and Provenance

> The Semantic Web isn't just about putting data on the Web. It is about making links, so that a person or machine can explore the Web of data. With Linked Data, when you have some of it, you can find other, related, data. — Berners-Lee (2006)

Berners-Lee (1999), the creator of the World Wide Web, described a vision of a Web where machines could communicate directly with other machines, swapping data and being able to act upon the information contained within, in much the same way as humans use the traditional Web to publish and browse documents. Berners-Lee et al. (2001) defines the Semantic Web as "an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation". The primary difference between the traditional World Wide Web — the Web of documents — and the Semantic Web — the Web of data — is structure. In order for machines to be able to understand data from other sources, it has to first

be given structure and meaning. This can either be done by using currently imperfect artificial intelligence to parse the free text of the pages of the traditional World Wide Web — an approach used to great success by search engines like Google (Brin, 1999) — or by using a more formalised method of annotating the data. The Semantic Web utilises the latter approach. In order to realise this vision the concept of Linked Data has been created, and in large part, it is beginning to be achieved through the Resource Description Framework (RDF).

2.1.1 The Resource Description Framework

As stated, the Resource Description Framework, or RDF, is one of the principle technologies by which the Semantic Web is being realised, and is a suite of recommendations maintained by the World Wide Web Consortium (W3C). Initially published in 2004 (Klyne et al., 2004), as of writing the most recent version is RDF 1.1 (Cyganiak et al., 2014). It is an abstract, graph-based way of representing information about resources. These resources are denoted by URIs — Uniform Resource Identifiers, initially defined in IETF RFC 1630 (Berners-Lee, 1994)[1]. URIs themselves formally contain no semantic information within the RDF model, serving purely as identifiers that should be handled opaquely when not being resolved (Berners-Lee, 1998).

The basic building block of RDF is the concept of a *triple*, which comprises a *subject*, a *predicate*, and an *object*. The subject and the object form two nodes of the graph, connected by the predicate as an edge. Predicates are directional, leading from the subject to the object, and consequently the resulting graph is also directional. The subject is either denoted by a URI or represented as a *blank node* — a specific, but unnamed node. Likewise, the object is either denoted by a URI or represented as a blank node, but can additionally be represented by a literal value. The predicate can only be denoted by a URI.

Because the primary goal of RDF is the interchange of data over a network, there are a number of different serialisations available. These range from those specifically created for RDF — Notation3 (Berners-Lee and Connolly, 2011), Turtle (Beckett et al., 2014), NTriples (Beckett, 2014), TriG (Bizer and Cyganiak, 2014) — to those adapted from existing formats — RDF/XML (Gandon et al., 2014), RDFa (Adida et al., 2015), RDF/JSON (Davis et al., 2013), JSON-LD (Sporny et al., 2014). In addition to numerous serialisation formats, there is also a powerful query language, SPARQL (Harris et al., 2013). SPARQL allows developers to write queries — not too dissimilar to SQL queries for relational databases — to be executed over a graph, or even a collection of graphs.

[1]It should, perhaps, be noted that in the latest versions of RDF, resources are actually denoted by Internationalized Resource Identifiers (a generalised form of URI), as defined by IETF RFC 3987. However, due to the familiarity of the term URI, this will be the term used in this work.

Similar to many relational databases, SPARQL also incorporates a protocol for querying graphs over a network.

In order to build interoperable knowledge-graphs, it is also necessary to have some core concepts in common, and this core vocabulary is provided by the RDF-Schema (RDFS), which is part of the suite of RDF recommendations. Nevertheless, the Web Ontology Language (OWL) (W3C OWL Working Group, 2012) was created to be used as a common way for systems to map ontologies firstly onto RDF, and by association onto RDF serialisation formats, thus allowing for conceptual models to be shared between systems. As RDF has become more widely used, developers have coalesced around a number of common ontologies (for example, FOAF (Brickly and Miller, 2014), Dublin Core (DCMI Usage Board, 2012)). There is a clear motive for reusing these ontologies, as they help developers to more easily integrate data from other sources into their systems, as well as helping to make their data available to other systems which might want to use it. Nevertheless, their use is not mandatory, and developers are free to use alternatives or create their own conceptual models if they wish. PROV-O, which will be discussed in more detail in Section 2.1.3.2, is an example of an ontology written in OWL.

It should be stated that Linked Data is not limited solely to RDF and SPARQL, with formats like JSON being used more and more frequently. This is particularly true with the introduction of JSON-LD Sporny et al. (2014) which shares many of the concepts and features of RDF, but packages it in a JSON format that developers are more likely to be familiar with, and which already has almost universal support both client-side with Web browsers and server-side with most modern programming languages.

2.1.2 Provenance in the digital age

As the Semantic Web is built on many of the same principles as the traditional Web — in particular, the fact that people should be free, insofar as the technology is concerned, to publish anything they want — issues have arisen concerning the accuracy or reliability of anything published on the Semantic Web. The solution to many of these issues is to record the history of the data; how it came to be; for what period it can be considered valid; who was involved in its creation; what it was used for; and so on. This history has come to be known as *provenance*. For the purposes of this work, the definition of provenance from the PROV-DM recommendation shall be adopted (see Section 2.1.3.1):

> Provenance is defined as a record that describes the people, institutions, entities, and activities involved in producing, influencing, or delivering a piece of data or a thing. — Moreau and Missier (2013a)

The study of provenance within computer science, however, stretches back to before the creation of the Semantic Web into the fields of databases and e-science. Consequently,

a review by Moreau (2010) demonstrated that a number of the more influential papers within the provenance community stem from these fields (e.g. Buneman et al. (2001), Simmhan et al. (2005)). In both these cases, being able to demonstrate how a particular result was generated (whether as a result of a database query or an experiment) is seen as valuable, adding to the trustworthiness of the data being presented. Even beyond the field of computer science, the editor of *Science*, McNutt (2015) recently noted that while scientists might have different interpretations of data, as long as the data is published alongside the researcher's methodology and analysis, there is an opportunity for critical, independent critique and reproducibility. Whilst she did not address digital provenance directly, it is clear that there is a desire to have such a capability within the wider scientific community. Similarly, a recent symposium, reported in Burgess et al. (2016), was dedicated to understanding reproducibility in data-intensive fields of research, whilst Corsar et al. (2016) discuss the importance of reproducibility in the social sciences, in particular with respect to social media data.

Additionally, beyond the world of the Web, there are a number of areas where people are wanting to track provenance digitally. One area of interest is tracking supply chains, which in our globalised economy have become tortuously complex. Take, for example, your food: whether it is the supply chain providing food for human consumption (Moreau, 2013), or the processes involved in handling it (Markovic et al., 2014), some believe that provenance has the potential to bolster consumer confidence in food and potentially even improve health outcomes.

During the process of preparing to standardise provenance on the Web, it was identified that there were many different active areas of research within the provenance community (Moreau, 2010). However, many of these topics are beyond the scope of this work, and will not be discussed here. Instead, as the focus of this work is to extend the field of natural language generation to enable users to understand the provenance information stored in the Semantic Web, the remainder of this section will focus on the development of provenance for the Semantic Web.

The primary distinguishing feature of provenance on the Semantic Web — as opposed to database provenance, for example — is its potentially highly decentralised nature. If, for the moment, it is assumed that Berners-Lee's vision of a highly interconnected Web of data has been realised, then one can easily imagine a scenario where the generation of a single piece of data — for example, a predicted temperature at a particular time and place — might involve data from hundreds of machines under diverse control (reports from many privately-owned internet-connected weather stations, weather satellite images, Met Office models, Ordnance Survey maps, etc.), where each of those pieces of data potentially has provenance of its own (sensor calibration, image post-processing, model source data, etc.). Taking this into account, interoperability between provenance-enabled systems on the Semantic Web was clearly an issue that would need to be tackled for this vision to become a reality.

Because of this clear need for the interchange of provenance on the Web, a number of ontologies were created, for example: Provenir (Sahoo and Sheth, 2009), the Provenance Vocabulary (Hartig and Zhao, 2010), and the Open Provenance Model (Moreau et al., 2008b, 2011). The Open Provenance Model was formalised following three provenance challenges (Moreau et al., 2008a; Simmhan et al., 2011), exploring exactly how provenance could be exchanged on the Web. Sahoo et al. (2010) provided informed mappings between a number of these early provenance representations (including some ontologies not specifically intended to be used to record provenance, but with potential overlaps) and the Open Provenance Model (OPM), which at the time was being considered as the basis for a standardisation effort within the W3C. The W3C Provenance Working Group published its suite of PROV recommendations in 2013 (Moreau and Missier, 2013a).

As is perhaps to be expected with standards, and was always intended with PROV, there have been a number of innovations that have led to extensions to PROV since its promulgation, helping to expand its usefulness within specific application domains (e.g. Markovic et al. (2014); Cuevas-Vicenttin et al. (2014); Missier et al. (2013), etc.). It is possible that at some time in the future it may be necessary to have a second standardisation process, incorporating these developments. However, this work will focus solely on the original PROV recommendations, without any extensions, so as to avoid unnecessary complication.

2.1.3 PROV — Provenance standardised on the Web

PROV, as standardised by the W3C, is a collection of four recommendations (PROV-DM, PROV-N, PROV-O, and PROV-CONSTRAINTS), along with a number of working group notes. PROV-DM is the PROV data model, and describes the abstract concepts and relations used in PROV. PROV-N is an ostensibly human-readable syntactic notation for PROV, used primarily to communicate these provenance concepts and relations between PROV experts or between experts and PROV-enabled computer systems. PROV-O is the OWL ontology used to map the concepts of the PROV data model onto RDF. PROV-CONSTRAINTS defines the inferences one can make from a provenance graph, as well as the constraints that determine relationships that would render a PROV graph invalid. Of principal interest for this work are PROV-DM, PROV-O, and PROV-CONSTRAINTS, as PROV-N describes a way of representing PROV that is not accessible to casual human users, as its primary purpose was to help define the formal semantics of PROV. Both PROV-N and PROV-O use URIs to denote provenance resources, with PROV-O also using them to denote the relations between those resources.

2.1.3.1 PROV-DM

PROV-DM (Moreau and Missier, 2013a) is the PROV data model, and describes the abstract model of provenance used by the other recommendations and working group notes. PROV is based upon three core concepts: *Entities*, which are data, or physical things; *Activities*, which are events that occur; and *Agents*, which are involved in a situation, and can be modelled as either entities or activities depending on the context of the application. Additionally, a number of relations are defined, which connect instances of each of these concepts. Together, these concepts and relations help describe the lifetime of a thing or a piece of data, and provide a way in common for provenance-enabled systems on the Web to conceptualise the provenance when sharing this information. PROV-DM does not provide, however, a way of representing these concepts, leaving that task to the PROV-N and PROV-O recommendations.

As an example of some of the relations available in PROV-DM, an activity can be said to have *used* an entity, whilst an entity can similarly be said to have been *generated* by an activity. Some of the relations are more complex and can involve more than two instances of the core concepts. For example, when talking about *derivation*, it is possible to state that an entity was derived from another entity, by an activity. Consequently, the relations in PROV-DM are n-ary relations. Figure 1.1 shows the core concepts and their relationships, additionally demonstrating the directional nature of the relations. For a diagrammatic example of one of the n-ary relations, see the Derivation in Figure 1.2. However, neither of these diagrams represents all the relations available within PROV-DM.

2.1.3.2 PROV-O

PROV-O (Lebo et al., 2013) is the OWL specification of the PROV data model — it provides the mapping between PROV-DM and RDF, bridging the gap between the abstract provenance concepts, and the core technology on which the Semantic Web is founded. In practice, PROV data transmitted between systems on the Semantic Web is most likely to take the form of an RDF serialisation using PROV-O.

This recommendation principally differs from PROV-DM in that RDF only permits binary predicates (i.e. a predicate describes a relationship between two resources), whilst, as has been previously asserted, PROV-DM describes a number of higher-arity relations. PROV-O's solution to this is to introduce the concept of a directed *qualified relation*. Qualified relations are the reified form of the relationship, where the n-ary relationship is represented as a single resource, with binary predicates to or from each of the other related resources. For example, to say that an activity used an entity at a given time, a PROV-O representation would state that the activity had a qualified usage, and that

PROV-N:
```
wasGeneratedBy(:gen; :ent, :act, time)
```

PROV-O:
```
:ent a prov:Entity ;
prov:wasGeneratedBy :act ;
prov:qualifiedGeneration :gen .

:gen a prov:Generation ;
prov:activity :act ;
prov:atTime time .

act a prov:Activity .
```

Figure 2.1: A comparison of the n-ary wasGeneratedBy statement from PROV-N and the reified version from PROV-O in the Turtle RDF serialisation format.

the qualified usage had properties relating it to both an entity and a time. This can be seen more clearly in Figure 2.1.

2.1.3.3 PROV-CONSTRAINTS

PROV-CONSTRAINTS (Cheney et al., 2013) helps ensure that provenance graphs in PROV are logically consistent. It details the inferences one can make from PROV-compliant provenance graphs, as well as describing the constraints necessarily satisfied for a provenance record to be considered as valid. This recommendation is of interest to this work, because the logical constraints in the provenance graph have a strong impact on the structure of the explanations that would be generated. In particular, PROV-CONSTRAINTS defines the temporal constraints of the PROV-DM, which provide a partial ordering of events within the graph, and which will be of importance when trying to determine how to structure the explanation. This is due to the fact that chronology is likely to be an important factor when trying to determine an appropriate structure for a natural language explanation. Additionally, the validity constraints help to discourage systems developers from generating PROV graphs that contain certain logical inconsistencies that would make them difficult or impossible to render sensibly in natural language.

To provide concrete examples, Inference 13 (attribution-inference) (Figure 2.2) states that if an entity is *attributed to* an agent, then it can be inferred that there is an activity *associated with* that agent, which *generated* that entity.

```
IF wasAttributedTo(_att; e,ag,_attrs) THEN there exist
a, _t, _gen, _assoc, _pl, such that wasGeneratedBy(_gen;
e,a,_t,[]) and wasAssociatedWith(_assoc; a,ag,_pl,[]).
```

Figure 2.2: PROV-CONSTRAINTS: Inference 13 (attribution-inference)

It is important to note the difference between the recommendation's inferences and constraints: constraints need to be satisfied in order for the graph to be considered valid, whereas inferences consist of additional PROV statements that can be considered true, provided certain inference requirements are satisfied, even if those statements are not explicitly represented in the graph. By contrast, an example of a constraint would be Constraint 55 (entity-activity-disjoint) (Figure 2.3), which states that if a resource is both an entity and an activity, then the graph is invalid.

```
IF 'entity' ∈ typeOf(id) AND 'activity' ∈ typeOf(id) THEN
INVALID.
```

Figure 2.3: PROV-CONSTRAINTS: Constraint 55 (entity-activity-disjoint)

A result of this constraint is that it is not possible for an agent to be conceptualised as both an agent and an activity within a PROV graph, but as is noted in the recommendation's remarks about this constraint, it is acceptable for an agent to conceptualised as one or other, or neither, of those two concepts.

2.1.4 Interfaces to PROV provenance

In the future, there may be many instances when it might be necessary for a human to make an informed decision based on a provenance graph on the Web. In these cases, it would be necessary for them to be presented with the provenance of that data in a way that best suits their capabilities, without overwhelming them, whilst allowing them to take maximal advantage from it.

As humans, there are two primary ways we communicate information to one another: diagrammatically and linguistically. (Body language is of course a major form of communication between humans, but as of yet, is not widely used as a form of communication with machines.) There exist a number of techniques for communicating provenance diagrammatically (for example, Figure 1.1 demonstrates one common approach to representing provenance, with entities, agents, and activities represented in different colours and shapes). Other examples are work by Toniolo et al. (2014) and Hoekstra and Groth (2015), which both use graphics as part of their interface for querying provenance datasets interactively.

On the other hand, there are few techniques for explaining provenance linguistically. One example of a system that attempts to do this is by Packer and Moreau (2015). However, their system uses string-substitution — a process where placeholders in a string are replaced by the value of variables — in such a way as to render their system application-specific. (This will be discussed in greater detail in Chapter 3.) As some application contexts would favour either linguistic or diagrammatic forms of representation, it is potentially useful to develop both forms of interface. This work endeavours to address the lack of a linguistic capability, in an application-agnostic way.

2.2 Natural Language Generation

Natural Language Generation (NLG) is the process by which an abstract communicative goal is computationally realised in the form of natural language. This is an important problem to solve in the field of artificial intelligence, because natural language is often the best way to communicate with a human. Whilst graphical interfaces can be very powerful ways of providing an interface for users, and are the primary way most able users interact with many of their devices, it is important to have an alternative for when such an interface is less appropriate or desirable. In the nascent market of digital home assistants, such as the Amazon Echo's Alexa, Apple's Siri, or Microsoft's Cortana, it is not hard to imagine that the quality of the natural language interface will be a strong factor determining the success or failure of certain products — with more complex natural language generation technology potentially providing that much needed edge over their competitors.

To provide a concrete example of why one might wish to present provenance information using natural language, one need only look to the original motivation for this research. This work was funded by a grant as part of the International Technology Alliance in Network and Information Sciences of the United Kingdom's Defence Science and Technology Laboratory and the United States' Army Research Laboratory. In a defence scenario, data can often pass through many different hands and processes on its path from sensor to decision-maker. By the time the data makes its way to an officer responsible for making decisions, it has often been distilled into a written report or is presented verbally in person or over a communications channel. Within this field, there is an interest in being able to automatically generate a provenance trace to form an intrinsic part of such reports, and as these reports already typically take the form of natural language, it is natural for these provenance traces to also take a linguistic form. The ultimate goal of such a technology would be to allow commanders to have a more appropriate level of confidence in the reports they read, and a greater degree of confidence in the decisions that they make.

For the purposes of this work, natural languages are considered to be languages that have not been deliberately engineered to have certain features, but rather have developed as part of a natural process over many years. NLG can take several forms, depending on the purpose of the systems of which it is a part. For example, a system designed to elicit knowledge from a user would be tailored to ask questions in a way as to encourage the user to provide an unambiguous answer containing the desired information. On the other hand, a system designed to explain something to a user needs to be able to use language in such a way as to help the user build a mental model of the thing being described. As the goal of this work is to explore the role of NLG for the purpose of generating natural language explanations from provenance, the rest of this section will be focussed on the state-of-the-art with regard to explanation systems, though this will

also require a discussion of more general aspects common to the design of all NLG systems. This section concludes with a discussion of the various methods previously used for the evaluation of NLG systems.

2.2.1 NLG Architecture

There are a number of proposed architectures for the design of NLG systems, however, one particular architecture that has gained popularity in the NLG community (Bateman and Zock, 2003) is the architecture initially presented in Reiter (1994) and further developed in Reiter and Dale (2000). This architecture was not initially intended to be prescriptive, but rather to describe the features common to the extant NLG systems, and consequently it is useful in helping to frame a discussion surrounding the research problems involved in using NLG technologies to generate provenance explanations. The name "consensus architecture" has come to be used (Power et al., 2003; Rambow et al., 2001) to talk of this architecture, and is the one that shall be used here.

Before going on to discuss the consensus architecture in more detail, it is important to acknowledge that there have been other attempts to systematise the design of NLG systems. For example, after reviewing a number of different extant NLG systems, Cahill et al. (1999) and Cahill et al. (2000) proposed an architecture called RAGS — Reference Architecture for Generation Systems — which was intended to be truer to reality than the consensus architecture and to better enable interoperability between system components created by different researchers. The architecture was later refined in Mellish et al. (2006). However, perhaps due to the complexity of the proposed architecture, RAGS has seen little, if any, adoption outside of original project, with perhaps only one published implementation (Cahill et al., 2001). By contrast, the more abstract consensus architecture seems to have a greater degree of adoption, perhaps due to the fact that it was intended as descriptive rather than prescriptive, and can consequently be tailored to suit the specific needs of a particular application (Reiter, 2007). This particular aspect of the consensus architecture will be exploited in Chapter 3.

The consensus architecture takes a pipelined approach to natural language generation, with three core modules each performing a number of tasks. The advantage of such an approach is that, provided the interfaces between modules are well specified, the modules can be treated as layers in a stack, and each layer can be developed individually, and interchanged easily. This is particularly useful for explanation generation systems, where the main research and development effort is focussed on the document planning (D) and microplanning (M) layers, whilst typically using simple or existing state-of-the-art systems for realisation (R)(Lester and Porter, 1997).

Figure 2.4 demonstrates the consensus architecture pipeline model of NLG, starting with an abstract communicative goal, and ending with a text ready for presentation.

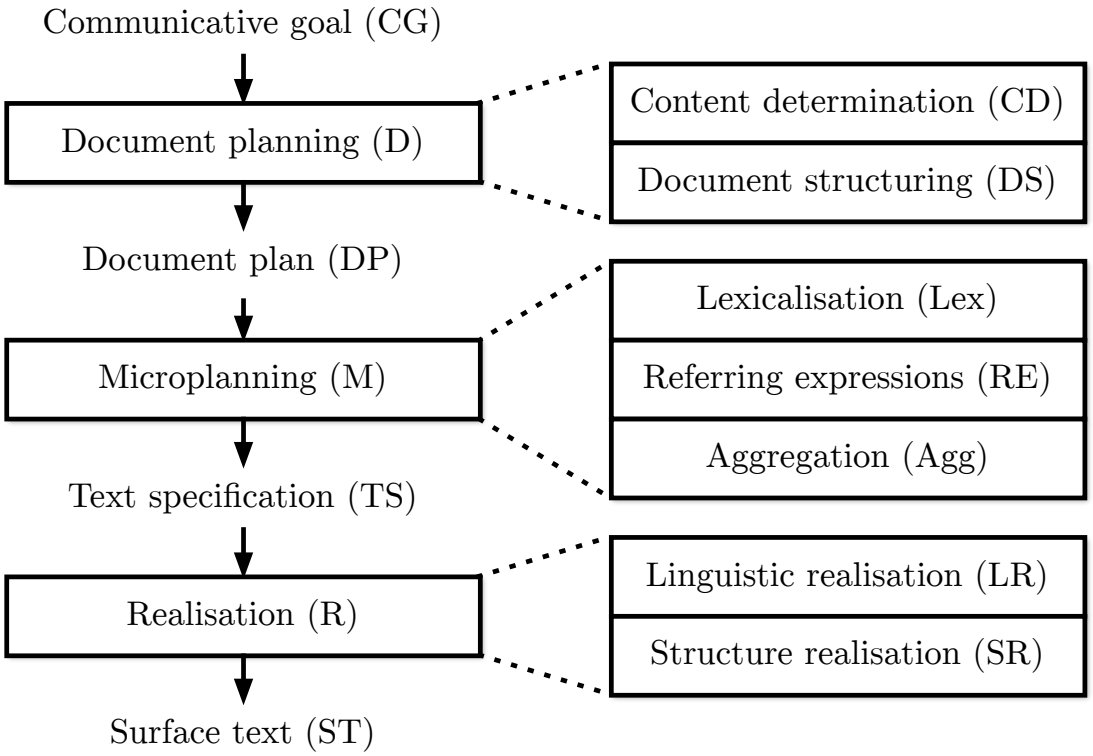


Figure 2.4: The consensus architecture of NLG, adapted from Reiter and Dale (2000, pp. 49 & 60)

Whilst, in the consensus architecture, there is no opportunity for feedback between modules, there can be a higher degree of interaction between the component functions of each module, where appropriate. In template-based realisers (see Section 2.2.1.3), for example, it would not be unusual for a significant amount of the structure realisation as well as the linguistic realisation to be achieved at the same time using the templates.

2.2.1.1 Document Planning (D)

In the document planning module, the abstract communicative goal (CG) — that is, the intended purpose of the speech act (Searle, 1969), e.g. to inform, to persuade, or to evoke a specific emotion or response, etc. — is used to produce a document plan (DP). In the work presented in this dissertation, the communicative goal (CG) is to inform a user of the contents of a provenance graph. A document plan is typically an ordered tree representing the various concepts that need to be expressed in the document and describing the flow of the text. A technique often used to develop this is Rhetorical Structure Theory (RST) by Mann and Thompson (1988) that describes a number of relationships that can exist between different parts of a text. One of the goals of RST is to be able to introduce the notion of *coherence*. If a text adheres to all of the constraints laid down by RST, then according to the theory it can be described as coherent, meaning

that the text should, at least with respect to itself, be consistent in its arguments and assertions. Because RST applies at the level of the document plan, before microplanning or realisation, a text that is coherent according to RST can still consist of nonsense. An example of a text that might be deemed coherent by RST, but is still largely nonsense, would be Lewis Carroll's *Jabberwocky*.

Whilst RST was designed to analyse human-generated texts for coherence, it is useful for NLG because systems that use RST to constrain their document plans are guaranteed to produce texts that are coherent, at least within the definition of coherency given. This has led to RST being widely used in the NLG community (Bateman and Zock, 2003; Reiter and Dale, 2000).

Content Determination (CD) The document planning module (D) can be characterised (Reiter and Dale, 2000, p. 63) as a function operating on the four-tuple $\{k, c, u, d\}$, where: k is the knowledge base available to the NLG system; c is the communicative goal; u is the system's model of the user; and d is the system's representation of the discourse with the user up to the point of execution. Based on this information, the document planner needs to be able to decide what subset of k needs to be communicated with the user represented by u to achieve the communicative goal c, given the context provided by d. This process is called content determination (CD), and produces a subset k' of k, which is, at this stage, still just an abstract knowledge graph.

For example, a student might want to tell their supervisor that they have not completed their work on time, but in doing so, they would also want to keep their supervisor from getting angry. In this case, it would be necessary to communicate not only the fact that the work has not been completed, but also other information to placate the supervisor, such as excuses they would consider valid, or assurances that the work is forthcoming. It should be clear that achieving this goal (c) depends not only on the information the student has available (k), but also their knowledge (u) of their supervisor, taking into account previous interactions (d).

Document Structuring (DS) However, it is often not sufficient to simply provide a user with information to achieve a communicative goal. If, for example, the goal is to convince the user that a specific course of action is the most appropriate, given a situation, then a well structured argument is called for. RST can play a part in this process, which is called document structuring. The process of document structuring results in a document plan (DP), which is an abstract ordered tree structure, representing the higher-level flows of the text, which is then passed to the microplanning module. A tree structure is used, as opposed to a linear structure, because it concerns the outline of the document, which is typically conceptualised as a tree (see, for example, the contents page of this dissertation).

2.2.1.2 Microplanning (M)

Microplanning is the process of going from a very high-level abstract representation (DP) of the structure of a text to a lower-level, though still largely abstract, representation of a text — called the text specification (TS) — that describes, in addition to the overall structure of the text, the content of individual sentences; the word roots — or lexemes — that are going to be used; the conjunctions that will be used to structure sentences; and other information that the realiser will find useful in realising the surface text.

Lexicalisation (Lex) Lexicalisation is the process of choosing which words or phrases to use to describe particular concepts. As an example, the phrase "the 44th President of the United States of America" uniquely identifies one person, and will continue to do so. In practice, however, this phrase is unwieldy, and potentially confusing: somebody who does not know that the 44th President of the United States of America is — at the time of writing — also the current President of the United States of America, might fail to make inferences necessary to understand a situation being described. It is far more likely that he would, at least until the end of his term in office, be referred to as "the president" or simply "Obama". This type of decision also needs to be made based on an understanding of the people with whom the discourse is taking place, where, extending the previous example, a Frenchman hearing the term "president" might mistakenly take it to mean the French president instead if enough context is not provided. This stage of generation is a particular focus of the work described in this dissertation, and is consequently discussed in more detail in Section 2.3.

Referring Expression Generation (RE) Referring expression generation has two primary purposes. The first, in the case where no appropriate word or phrase in the lexicon can be found to describe an object – perhaps its name is unknown, or the object is indefinite – and the system needs to refer to it anyway. As an example "the gentleman to your left" is a referring expression that does not use a particular unambiguous identifier to describe "the gentleman", but rather uses properties from the situational context to make it clear who is being referred to. The second purpose encompasses the use of linguistic features such as pronouns or anaphora to increase fluency, where appropriate, ideally without introducing ambiguity.

Aggregation (Agg) Aggregation is the process by which the abstract structures in the document plan are mapped on to high level linguistic features, such as paragraphs, sentences, and clauses. This includes using linguistic knowledge to plan when to use conjunctions to increase fluency by, perhaps, combining a number of short sentences, into one longer, more comprehensible sentence.

2.2.1.3 Realisation (R)

The realiser takes as its input the text specification (TS) produced by the microplanner, and produces as its output the surface text (ST) ready to be presented to the user. The surface text is not only, hopefully, orthographically correct natural language, but will also contain the structural markers — for instance, markup tags — required by the presentation system that is being used, such as a Web browser, for example. The process of converting the abstract text specification into orthographically correct natural language is called Linguistic Realisation (LR), whereas the insertion of structural markers is called Structural Realisation. The process of structural realisation (SR) can range in complexity, from the simplest being plain-text systems where the only tags needed are line breaks between paragraphs, to perhaps one of the more complex being HTML where markup tags are needed for both style and structure including headings, paragraphs, lists, and potentially even hyperlinks.

There are two main strategies for converting the text specification, which is an abstract data-structure, to surface text. The first of these (Hovy, 1987) is to use sophisticated grammars that use their knowledge of the natural language in question to build up sentences from the lexemes and referring expressions in the text specification, inserting punctuation and structural markers where necessary. The second approach is to use templates (Reiter and Dale, 1997) that match certain patterns in the text specification, and use variable-based string-substitution combined with a knowledge of certain language features — such as conjugation and number agreement — to produce the surface text. Grammar-based realisers are more sophisticated, and they are considerably more generalisable than those using template-based approaches. However, they are considerably more complicated to build, as they need to encode a relatively complete knowledge of how a language is used in order to function. Even so, van Deemter et al. (2005) express the opinion that there is little difference between the two approaches in terms of the quality of the texts produced.

2.2.2 Evaluation of Explanation Generation Systems

Mellish and Dale (1998) describe three broad perspectives on evaluating NLG. The first of these is about evaluating the properties of the NLG theory, and as the purpose of the work presented in this dissertation is not to extend NLG theory, but rather to demonstrate its role in helping users understand PROV graphs, this perspective will not be discussed any further.

The second perspective considers the potential for application of the technology, with factors such as feasibility, cost, and a comparison to other systems that could achieve a similar goal. Again, as the focus of the work presented in this dissertation is primarily on

whether such a system is even possible rather than the practical details of how it might be applied outside academia, this perspective shall also not be discussed any further.

The final perspective is concerned with evaluating properties of the NLG system, such as grammatical correctness, semantic unambiguity, fluency, or ability to help a user perform a specific task. A number of strategies for evaluating NLG systems from this perspective have been described in the literature, and the rest of this section will provide an overview of the most commonly used.

2.2.2.1 Direct Human Evaluation

In direct human evaluation, study participants are presented with a computer-generated piece of text, and simply asked to evaluate it, either qualitatively or quantitatively, depending on the study design. This can be a useful first-step in evaluation, for example in the medical domain, where domain experts will be able to quickly identify significant, and potentially dangerous, errors in the generated text. However, in many cases it is desirable to compare a text to a text generated by a previous generation of software, or by experts, in which case a blind comparison might be used instead.

2.2.2.2 Blind Comparison Evaluation

The blind comparison evaluation is, in effect, a variation upon the Turing Test. A study participant, in some cases domain experts (Lester and Porter, 1997), in others somebody similar to the intended user or consumer of the text, is presented with two documents. One of these documents is generated by the NLG system, whilst the other is generated by a human. The participant, who does not know which, if either, of the documents is computer-generated, is asked to evaluate each text according to a number of metrics, either qualitatively or quantitatively, depending on the study design, and decide which suits the evaluation criteria the best.

This technique lends itself well to situations where human-generated texts are available, or easy to acquire, and if the performance of the NLG system relative to humans is of interest to the study designer. It is, however, a less viable option for situations where human-generated texts are harder to obtain, or where only absolute performance is of interest.

2.2.2.3 In-context task evaluation

For in-context task evaluation, the system is evaluated in terms of how well it helps its users perform certain tasks. Usually this form of evaluation is most appropriate when NLG is being introduced to a system with some specific goal in mind. Reiter

et al. (2003), for example, presented a system designed to help people quit smoking by generating letters advising them how to quit, based on their responses to a questionnaire. They evaluated the system's efficacy through the use of a standard clinical trial assessing whether the users were actually more successful at quitting with personalised versus non-personalised interventions.

One aspect of in-context task evaluation is that because the NLG is embedded within a larger system with its own performance characteristics, it can be difficult to control some of the variables, however Reiter (2011) argues that the increased external validity of such a study is of greater concern to most NLG developers. That is to say, that as most NLG systems are designed for a particular application, the quality of the texts produced is more meaningfully evaluated with respect to how they affect the system's overall performance, rather than assessing them in isolation from the system and its context.

2.3 Generating natural language for the Semantic Web

As it involves the task of generating text from data to meet a particular communicative goal, Natural Language Generation is a natural fit for use with Semantic Web technologies. In fact, in both 2015 and 2016, there was a workshop on *Natural Language Generation and the Semantic Web* colocated with the annual *International Natural Language Generation* conference. Most of these papers, however, appear to focus on the use of large Web-based corpora to learn representations that could be used for NLG (Basile, 2016; Sleimi and Gardent, 2016; Peter et al., 2016), or addressed NLG in languages other than English (Sanby et al., 2016; Nesterenko, 2016), and are therefore not particularly relevant to this work.

In 2017 this workshop was replaced by a challenge (Colin et al., 2016) attempting to generate sentences from triples extracted from DBpedia. Of the six submissions to that challenge, three utilised a neural network based approach. With the advent of Deep Learning, and the size of the DBpedia corpus, this is not entirely surprising. However, as there are no suitably large parallel corpora of PROV data and corresponding explanations, this approach is not possible, and therefore not considered further. It should still be noted, however, that as techniques such as word embeddings (Mikolov et al., 2013) and Long- short-term memory units (Hochreiter and Schmidhuber, 1997) have recently been shown to be particularly effective at natural language based tasks, it would not be entirely surprising if this field becomes dominated by neural approaches in the future. Whilst not usually published at natural language generation venues, systems like DenseCap from Johnson et al. (2016) use Deep Neural Networks to generate natural language captions for a very wide range of images, with similar systems applied to generating descriptions of video (Venugopalan et al., 2015).

A review by Bouayad-Agha et al. (2014) demonstrates the full breadth of NLG research with respect to the Semantic Web. As part of their review, they list what they consider to be the biggest challenges still facing the NLG community with respect to the Semantic Web, listing domain portability as the number one challenge. By domain portability, they refer to what this work has been calling domain-independent or domain-agnostic generation; allowing a system to be applied to a new domain or dataset with minimal, or ideally no additional effort. This work aims to partially address this challenge.

The Semantic Web tasks that NLG technology has been applied to can fall into two categories: generating texts from Semantic Web data; using NLG and NLP as a bidirectional interface to edit data.

One of the first researchers to tackle the first challenge was Bontcheva (2005), who created the ONTOSUM system. This system was able to generate very basic sentences from RDF data. However, it was unable to perform any aggregation, and required a lexicon for every ontology it used. Since then, a number of approaches have been proposed for solving this lexicalisation problem, which is discussed in greater detail in Section 2.3. More recently, systems by Bouttaz et al. (2011) have attempted to generate natural language explanations of provenance (specifically from OPM, the precursor to PROV), using policies to perform the task of content determination. This system also required a lexicon.

As for the second challenge, a number of approaches have been suggested to allow for the round-trip editing of ontologies and instance data using natural language. A number of the earlier bidirectional approaches suggest the use of restricted grammars (Bernstein and Kaufmann, 2006) or Controlled Natural Languages (Kuhn, 2008) such as Attempto Controlled English (Fuchs et al., 2008) or ITA Controlled English (Braines et al., 2013), as these helped ensure that the machine was able to parse the user's responses correctly, though this obviously depended on the user being able to use the controlled language correctly. A later approach by Hielkema (2010); Hielkema et al. (2007, 2008), was based on the WYSIWYM (What You See is What You Meant) approach by Power et al. (1998). This system allowed a user to craft ontologies as well as enter and edit metadata, without the need for a complicated parser. This was because the system used the generated texts to constrain the user's input, rather than allowing them to enter free text. A novel approach to the lexicalisation challenge was used, with the system attempting to build a lexicon as it was interacted with by the user.

2.3.1 The lexicalisation challenge

One particular hurdle that faces those attempting to generate natural language from Linked Data is that of lexicalisation. That is to say that one has to be able to map the semantic concepts represented by RDF onto lexemes (words and roots) in the natural

language that convey those concepts. For example, the URI "`http://www.w3.org/ns/prov#Entity`" identifies the concept in the PROV ontology that would be rendered in English as "Entity". However, there is no formal reason in the RDF specification why a resource denoted by that URI could not more appropriately be described by the word "Thing" or "European Swallow" as the URIs themselves formally carry no semantic information and are used purely for unambiguously referring to nodes and arcs in the semantic graph.

There are three primary approaches to mapping RDF concepts to lexemes described in the literature. The first, and notionally the simplest, is to explicitly represent the linguistic information using the RDF model. An example of an ontology that can be used to do this is *lemon* (McCrae et al., 2011), which maps concepts in an OWL ontology to their lexicalisations. The lemon ontology also allows for the mapping of the same lexeme to multiple senses (for example, for words with different meanings, or subtly different senses dependent upon context), and for the representation of the various forms a word can take (e.g. "Entity" becomes "Entities" when plural). Whilst this approach provides all the linguistic information needed to generate meaningful natural language descriptions of the Linked Data, the downside is that these lexical entries need to be created for every concept that might need to be expressed in natural language. If that problem might be tractable for some small domains, the additional challenge of then creating lexical entries for all the instances of each class would be an impossibly onerous task in all but the most trivial of cases. In the case addressed in the work described in this dissertation, PROV itself would be small enough to be able to reasonably annotate in this way, but because PROV can be used to annotate provenance in any domain, a system for generating explanations from PROV graphs needs to be able to handle concepts from any domain. Consequently, another approach is required.

The second approach takes advantage of the fact that the Semantic Web of Linked Data often lies in parallel to, and crucially cross-links to, the traditional Web of linked documents. DBpedia (Lehmann et al., 2015), for example, has a large amount of Linked Data that is mapped to natural language representations, in potentially more than one language, on Wikipedia. These large tagged natural language corpora allow for the automated extraction of lexemes for the ontologies used therein, and potentially reused elsewhere. Ell and Harth (2014) describe an approach to extract not just lexemes, but even NLG sentence templates from such corpora. However, not all applications are likely to have natural language texts associated with the Linked Data for this extraction to take place, limiting the usefulness of this approach.

The third and final approach to lexicalisation is to use the incidental, informal information encoded in the URIs of the Linked Data. That is, if the URI for a resource is "`http://example.net/people/John`", trusting that the resource is a "person" called "John", even if the formal RDF model does not allow us to make that inference with certainty. The assumption is that developers and ontology designers will use language

that adequately describes a concept. Obviously this only works when a linguistic URN is used, in contrast to a UUID or even potentially a blank node. This approach has been used in a number of cases (Sun, 2008; Powers and Stirtzinger, 2011), with Mellish and Sun (2006) describing a survey of OWL ontologies available on the Web investigating how ontologists use language when minting URIs for their classes and properties. After tokenising URIs, they used WordNet (Miller, 1995) to identify what types of words were used, primarily finding that 72% of classes have a URI ending in a noun, and 65% of properties have a URI beginning with a verb.

Developing upon that work, they built a system that was able to generate descriptions from small sets of triples (6 or fewer at a time) in a domain-generic fashion (Mellish and Sun, 2005; Sun and Mellish, 2006, 2007). However, primarily because of its domain- and application-generic approach, their system tended to generate relatively stilted or unusual English (Sun, 2008), and was only able to work when the triples contained the appropriate linguistic information with no fall-back mechanism for when this information was absent. Additionally, the Mellish and Sun (2006) study only examines ontologies and not instances of classes from ontologies. It would not be unreasonable to argue that ontologists put more care and attention into — if not agonising compulsively over — the precise language they choose to use when minting URIs, compared to developers of systems that create instances of classes from those ontologies to describe their data. Finally, ten years have passed since that study was conducted, and given the degree of change that has come about on the Semantic Web in that time, the results of the study may no longer hold true. Nevertheless, this approach shows enough promise to merit further research and, given that it would be the most suited lexicalisation approach for generating explanations of provenance from PROV, it consequently forms the basis for the approach described in later chapters.

2.4 NLG to generate explanations from PROV

There have been a number of attempts to apply NLG techniques to generate natural language explanations from PROV graphs. For example, Bouttaz et al. (2012) and Packer and Moreau (2015) both use templates and string-substitution to present provenance to users. In the former case, the templates created stuck closely to the PROV data model, and consequently were capable of generating explanations from any PROV graph, at the cost of using less expressive, natural-sounding English. The latter case, on the other hand, used templates specifically tailored for an application, which had the benefit of being more natural English, but were only useful in the context of that specific application.

Another example of exposing provenance to users using templates is the United States' 2014 National Climate Assessment[2]. Figure 2.5 shows the metadata about a particular figure from that report, including information about authorship and copyright. Near the bottom of this webpage is a sentence explaining which dataset the figure is derived from, and a link to information about the process that created it: "This figure was derived from *dataset U.S. Climate Divisional Dataset Version 2* using the activity *in-nca3-cddv2-r1-process*."

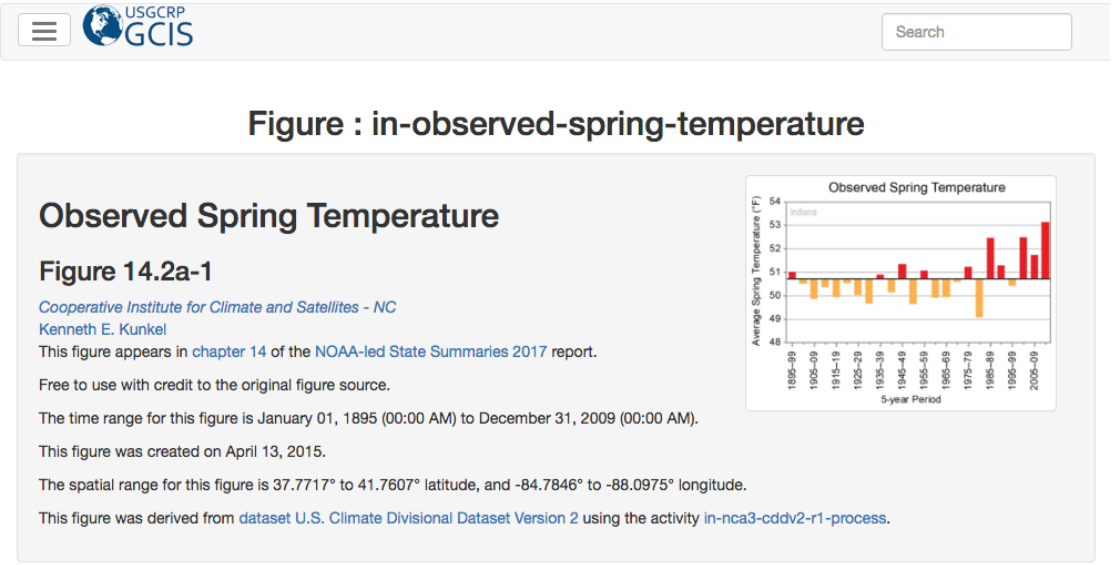


Figure 2.5: A screenshot from the 2014 National Climate Assessment, showing the metadata about a particular figure in the report, including a sentence automatically generated from PROV.

However, there are a number of features of PROV that offer interesting opportunities to be exploited by an explainer, whilst the challenge of requiring the use of only features required by the PROV specification precludes the use of many existing techniques, such as many of those discussed in Section 2.3.

PROV provides an interesting challenge for NLG because, unlike most other examples of RDF, there is a clear temporal ordering as well as a record of who did what. These can be used as the foundations for building a narrative-aware explanation system that does not simply systematically convert RDF into text, but rather tells the user the story of what has happened. This task is distinct from existing attempts to generated NL from Linked Data, which are typified by converting small amounts of RDF into one or two sentences, as in this case, depending on the quantity of input data, it might be necessary to generate considerably longer texts.

[2] `http://nca2014.globalchange.gov/`

2.5 Summary

In this chapter, a number of issues have been described regarding the generation of natural language explanations from provenance graphs. With increasing adoption of Linked Data technologies in general, and PROV in particular, it is increasingly necessary to have a way of expressing provenance linguistically.

There are a number of challenges to be addressed before this would be achievable, though the state of the art of NLG technology offers hope. These technologies, in particular the last approach to lexicalisation described in Section 2.3, combined with the particular features of the provenance explanation application, have the potential to work together to allow for natural sounding explanations of provenance to be created in a domain-generic way.

Chapter 3

An Architecture for Provenance Explanation Generation

In the previous chapter, the state-of-the-art with respect to the task of generating natural language was reviewed. This chapter presents the application of these technologies to the challenge of generating natural language explanations from abstract provenance data. After clarifying the scope of this work in Section 3.1, it begins, in Section 3.2, by describing a new template-based architecture for NLG that expands upon the existing consensus architecture described in Chapter 2. Section 3.3 describes an approach, not previously reported in the literature, to selecting the desired sentences from all the possible candidate sentences generated by these templates. Section 3.4 gives more detail as to how these templates work, and how their design informs the design choices for the architecture. Section 3.5 returns to the newly proposed architecture, explaining how the various modules differ from those of the consensus architecture, and provided justifications for those differences. This chapter concludes, in Section 3.6, with an example generation, helping to illustrate the operation of the various modules of this architecture.

The contribution of this chapter is the resulting architecture, which will be used in later chapters to generate explanations from provenance graphs in a domain-independent manner. This extension to the state-of-the-art allows for the generation of richer explanations of provenance, enabling the utilisation of mature off-the-shelf technologies for realisation, instead of simple string-substitution.

3.1 A clarification on scope

Chapter 2 introduced the consensus architecture for natural language generation, and described the three modules and seven processes involved in NLG. The template-based pipeline introduced in Section 3.2 is specifically tailored for generating natural language

explanations from PROV graphs in a domain-agnostic way. Particular attention is paid to a subset of the processes, some of which become the focus of Chapters 4 and 5, whilst the remaining processes are left for others. Some of these decisions in scope were taken because of practical timing constraints, whilst others were taken because it made for a better system to leave that work for others.

For example, it was not necessary to focus on realisation as a part of this work. The process of taking a text specification and producing a surface text is already a well-explored area, with SimpleNLG (Gatt and Reiter, 2009), in particular, often used to perform this task. Consequently, in this work, SimpleNLG is used to turn the text specifications into realised surface text. Attempting to tackle this part of the NLG pipeline as part of this work would no doubt have had serious negative implications for the quality of the sentences generated.

Similarly, this work does not focus any attention on content determination. As discussed in Chapter 2, this is the task of deciding which subset of the knowledge-base should be conveyed to a particular user, taking into account prior interactions and context, in order to achieve a particular communicative goal. The goal of this work is to achieve domain-agnostic generation of provenance explanations, and yet the communicative goal, the nature of the user, and the context of the communication are all very dependent on the domain and application. Consequently, it is assumed that the content determination stage of natural language generation will have to be built on an application-by-application, domain-by-domain, basis. Nevertheless, the modules that were implemented as part of this work, and described in greater detail in Section 3.2, mean that systems developers would only have to choose which parts of the PROV graph should be communicated to the user, needing no consideration of language at all.

Other aspects of the NLG pipeline received less focus because of practical constraints. For instance, no attention was paid to aggregation, as that is a somewhat complicated area in its own right, and deserved dedicated research rather being considered a simple add-on. As discussed in Chapter 6, this is left for further work. Similarly, whilst a basic algorithm for referring expression generation was implemented based on counting previous references to a resource, there was insufficient time to ensure it performed well, and it was, in fact, removed for the purposes of the experiments in Chapters 4 and 5.

This leaves the processes of lexicalisation and document structuring. These form the core of this work, and are in turn discussed in greater detail in Chapters 4 and 5, along with the experimentation that was used to inform and evaluate their development.

3.2 Template-based pipeline

Templates — where values are substituted into the variables of a predefined structure — offer a powerful way of generating natural language from Linked Data. They are much simpler to create than sophisticated grammar-based systems, and can easily be adapted to suit new applications and needs. However, unlike a more traditional approach to NLG, which would build up a text starting from a communicative goal, generating a document plan before then building independent sentences using something like RST, a template-based approach has a much more limited range of sentences that can be generated. In fact, for a specific provenance graph, there will be a limited, and relatively low, number of possible instantiations for each template — the total number of sentences a template-based system can generate is the sum of the number of possible instantiations for each template.

In contrast to the consensus architecture presented by Reiter and Dale (2000) and described in Chapter 2, the template-based architecture presented in this work does not maintain the traditional ordering of the processes involved in NLG — shown previously in Figure 2.4 — despite maintaining the core pipeline model, and recognising the importance of all the processes. Instead the proposed architecture, as shown in Figure 3.2, is chosen to allow for the NLG to be as computationally simple as possible, enabling the real-time generation of provenance explanations. As discussed in Section 3.1, not all the processes and modules are of equal concern to this work. The modules and processes shown in white are those that form the core of this work, with those in light grey of less direct interest, primarily left for others, and those in darker grey left for others in their entirety.

Before explaining in Section 3.5 what each stage of this pipeline achieves, and how it is able to use the templates, Sections 3.3 and 3.4 will take a close look at specific aspects of the problem which strongly impacted the design of the overall architecture. Section 3.3 describes how the problem of generating paragraphs from templated sentences can be conceptualised as a set-cover problem — when dealing with Linked Data — whilst Section 3.4 provides a detailed description of what form the templates used in this system take, which is heavily influenced by the choice to conceptualise this task as a set-cover problem.

3.3 Sentence selection (SS)

As stated, this system attempts to convert an entire PROV graph into natural language, assuming that any processing to remove unwanted content has been done prior to the system being invoked.

Consequently, it is possible to conceptualise the problem of converting the provenance graph into natural language using templates as a set-cover problem — that is, finding the subset of all possible sentences that can cover the set of all triples in the provenance graph. This is because we have a number of templates, each capable of producing a number of sentences from the graph. In turn, each of these sentences encodes the information from one or more triples in the graph. Because we do not want our explanations to repeat information unnecessarily, we want to find the minimal number of sentences that can be used to describe all the information in the graph. This is an approach to sentence selection and document planning not seen in the existing academic literature.

3.3.1 Selection as set-cover optimisation

To expand upon this notion, there exists in computer science the problem of set-cover optimisation (Karp, 1972). In this problem, there exists a set S that itself consists solely of sets. The set-cover optimisation problem is that of finding the lowest cardinality set M, where

$$M \subset S, \ \cup M = \cup S$$

The challenge of selecting sentences to build up a document from all the possible sentences than can be generated can be characterised as being broadly similar to the standard set-cover optimisation problem, though there are a number of other factors that have to be taken into account:

- Firstly, it may not be possible to cover all the triples in the graph because it is expected that there may be some relationships expressed in the RDF that the templates do not know how to handle, because they rely on ontologies other than PROV. As such the algorithm must be able to terminate successfully in such circumstances.
- There may not be an absolutely minimal solution in which every triple is expressed only once. This is likely to be the case, in particular, for triples that express the class of a resource (i.e. `(ex:Samuel a prov:Agent)`), as these triples will likely be included in the coverage set for a number of sentences involving that resource.

Taking those factors into account, it is still possible to use an algorithm used for the set-cover optimisation problem, with a number of small alterations. However, as the set-cover optimisation problem was shown by Karp (1972) to be NP-hard, it would therefore become computationally intractable with relatively small provenance graphs. Consequently, a greedy algorithm is used instead, guaranteeing timely termination, if a potentially sub-optimal solution as demonstrated by Johnson (1973).

The specific algorithm used, taking into account the factors listed above is described in Figure 3.1. The greedy behaviour is shown in Step 7, where the set of bindings with the largest cardinality coverage set is chosen in cases where there are no triples in the graph that can only be expressed by one set of bindings.

By virtue of the fact that each iteration removes at least one set of bindings from the bindings pool, and that the algorithm terminates when the pool is empty, the algorithm therefore satisfies the requirement that it always terminate.

3.4 Templates

Within the scope of this system, using the Sentence Selection algorithm from Section 3.3, a template comprises three core functions:

Bindings function *Input: graph; Output: sets of bindings* — A function that takes the provenance graph or subgraph and returns a list of sets of bindings, typically achieved by executing a SPARQL query over the graph, e.g.: `[{"thing":"ex:007", "type":"prov:Agent"}, ...]`. The bindings function can, however, be more complex, performing more aggregation, for example, that can be accomplished by a SPARQL query. Each set of bindings corresponds to a possible expression of a part of the graph in textual form — that is, a possible sentence. Because of the many possible combinations of sentences that could be used to express a graph, many more sets of bindings are generated than are actually necessary to build the document.

Coverage function *Input: graph, bindings; Output: coverage set* — A function that returns the set of triples in the provenance graph that can be inferred from, or are expressed by a sentence generated from a set of bindings, though these triples may not be explicitly represented in the sentence, e.g.: `[(ex:007, rdf:type, prov:Agent), ...]`. For example, if a sentence refers to some thing as a collection, it is possible to infer that the thing is also an entity (as prov:Collection is a subclass of prov:Entity). Consequently the triple stating that the thing is a prov:Entity would be included in the coverage set as well as that stating prov:Collection.

Text specification function *Input: bindings; Output: text specification* — A function that takes a set of bindings and returns the sentence as a text specification, ready to be passed to the realiser. This is done by generating the lexicalisations and referring expressions, expanding the bindings into the pre-existing text specification template where necessary. Unlike a simpler string-substitution method, by generating a text specification at this stage, instead of the surface text, it is possible to make use of an existing off-the-shelf realisation engine.

Inputs: Graph G, Templates T
Outputs: Set of sets of bindings to be expanded D

1. There is a graph, G, of triples to be transformed.
2. There is a set of templates, T, that can be used to transform triples into natural language.
3. There is an initially empty set of sentences chosen for the document, called the document set D.
4. For each template $t \in T$, generate all the sets of bindings, b— each set corresponding to a possible sentence — that can be generated from G. This is called the bindings pool, B.
5. For each set of bindings $b \in B$, calculate the subset of triples, $C_b \subset G$ that are expressed — that is, covered — by that template with those bindings. These subsets of triples, C_b are called coverage sets.
6. The union of all those coverage sets is the set of all triples that can be expressed by that set of templates from the complete graph $\cup\{C_1, C_2, \ldots, C_n\} = P \subset G$. This is called the triple pool, P.
7. Then:
   - If there are any triples $p \in P$ for which there exists only one $b \in B$ where $p \in C_b$, then that set of bindings must be added to the document set, D.
   - Else, the set of bindings $b \in B$ with the largest cardinality coverage set C_b is added to the document set, D, instead.
8. When any set of bindings, b, is added to the document set, D, b is removed from B and each triple $p \in C_b$, is removed from the triple pool, P. Additionally, where p does not take the form `(?x a ?y)`, any set of bindings whose associated coverage set has that triple as a member, is also removed from the bindings pool. If, at any point, removing a set of bindings from the bindings pool would leave a triple in the triple pool with no bindings that could cover it, then that set of bindings is also added to the document set.
9. Repeat Steps 7 to 8 until there are no more sets of bindings in the bindings pool.

Figure 3.1: The sentence selection algorithm

It is beyond the scope of this work to decide what provenance data should or should not be presented to a user — Content Determination (CD) — and instead this is left for future work. Rather the approach described attempts to convert an entire PROV graph into natural language, assuming that any processing to remove unwanted content has been done prior to the system being invoked.

In this system, it is the templates that define the syntactic shape of the sentences that are generated — that is, the structure of the sentences is, to a certain degree, hard coded into the Text Specification function of each template. This means that syntax realisation, which would normally occur in the Realisation (R) stage of generation, actually occurs when the set of bindings is generated from the graph, leading to the creation of the Template Matching process (TM) shown in Figure 3.2.

The process of document planning can now be divided into two parts: choosing which of the many overlapping template instantiations to use (Sentence Selection, SS), and deciding in which order to put them (Document Structuring, DS). The solution to the first of these challenges is described in Section 3.3, whereas the latter is explored in greater detail in Chapter 5.

Finally, the process of Microplanning from the consensus architecture is now achieved entirely when the templates are expanded (Template expansion, TE), with lexicalisation (Lex), referring expression generation (RE), and aggregation (Agg), all occurring at this stage. It is necessary for this expansion to happen after Sentence Selection (SS), because expansion can be resource intensive, and is wasted effort if performed on sentences that are not included in the final text.

3.5 Components of the template-based pipeline

As previously stated, a number of changes were made to the consensus architecture. These were made primarily in order to accommodate the use of templates, and the sentence selection algorithm from Section 3.3. This section will describe the modules and processes of the new template-based pipeline, highlighting the differences between the two architectures and justifying the changes.

The relative positions within the pipeline of all of these modules and processes can be seen in Figure 3.2, which compares with the consensus architecture from Chapter 2, as shown in Figure 2.4.

3.5.1 Content determination (CD)

This module performs exactly the same role as the content determination process from the consensus architecture, as described in Section 2.2.1.1. As stated in Section 3.1,

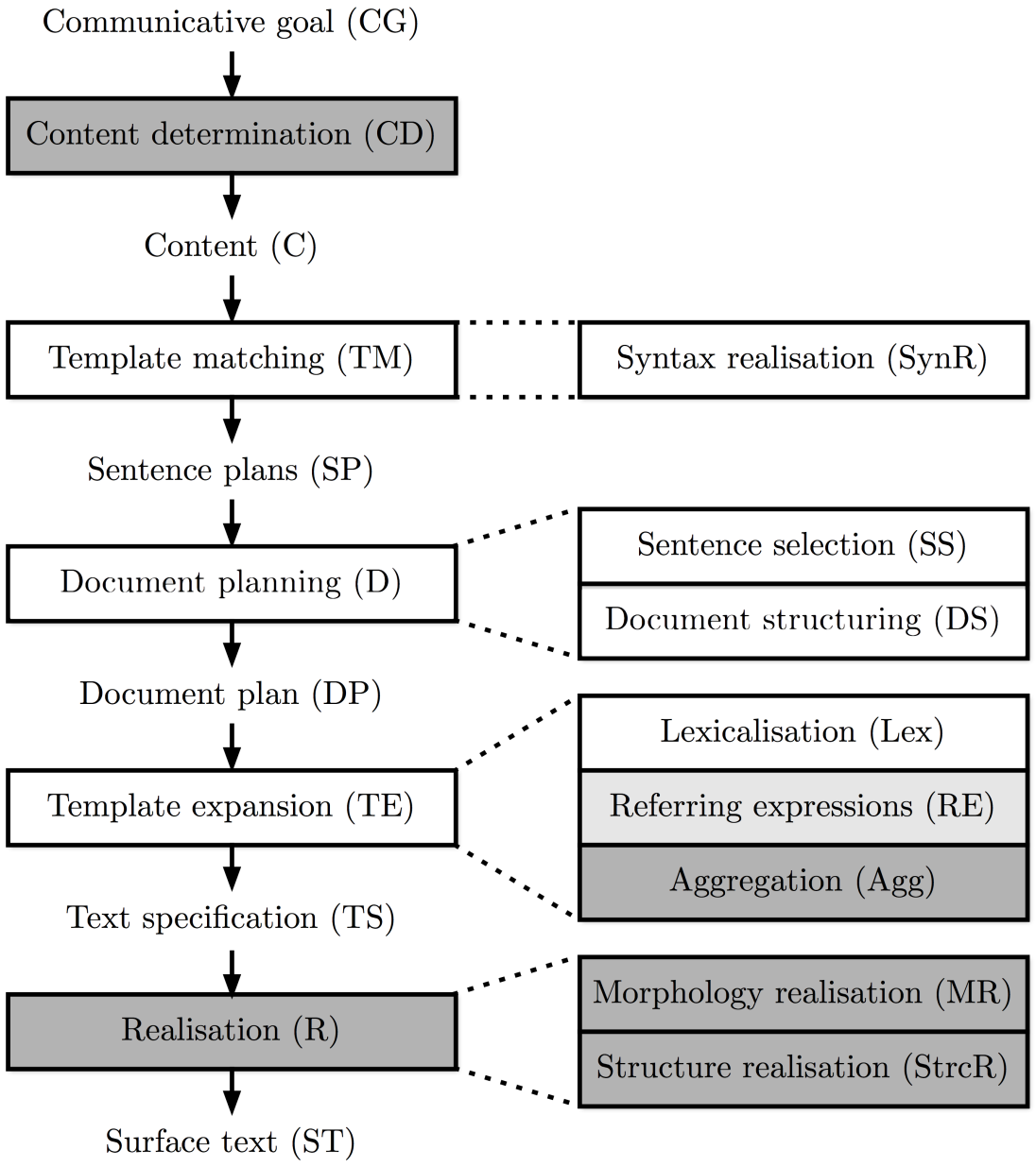


Figure 3.2: The template-based PROV explainer architecture described in this work, with the components forming the main focus of this work in white, and the components of less interest in light grey. The components in dark grey, whilst important to the architecture, are not of interest to this research effort.

this task is left for those developers who would integrate such a pipeline as this into their provenance-aware applications, due to its inevitably domain-specific nature. For the purposes of this work, it is assumed that the system should attempt to generate an explanation from the entire provenance graph with which it is presented.

3.5.2 Template matching (TM)

As its name might suggest, this is a new module for the template-based architecture. In terms of the consensus architecture, this could be considered as performing a similar role as part of the Microplanning module from Section 2.2.1.2. As described in greater detail in Section 3.4, each template has a number of associated functions, one of which is eventually used to produce a text specification. Whilst this specification is not generated until later in the pipeline, as additional stages need to completed first, once a template is matched, by executing its binding function, the syntactic nature of the sentence is largely predetermined. This stage of the pipeline potentially generates many more (template, bindings) pairs, constituting sentence plans, than are needed to produce a minimal explanation of the whole graph. In the consensus architecture this syntax realisation would typically be achieved by the realisation engine, but in the template-based architecture is coded into the templates.

3.5.3 Document planning (D)

Excepting the fact that content determination has been removed and placed earlier in the pipeline, the document planning module performs the same broad function as in the consensus architecture; its role is to decide the broad outline of the document or, in this case, the explanation. In the template-based architecture it does this in two stages: sentence selection, and document structuring.

Sentence selection is the process whereby the many redundant (template, bindings) pairs generated in the template matching stage are reduced down to, ideally, a minimal set that would adequately cover the whole graph. This process is described in greater detail in Section 3.3, and results in an unordered set of (template, bindings) pairs.

Document structuring, in the template-based pipeline, is the process of choosing an order for the sentences that have been selected. This component is the focus of the experiments in Chapter 5.

3.5.4 Template expansion (TE)

This module shares the same subprocesses as the Microplanning module from the consensus architecture. These subprocesses are lexicalisation, referring expression generation,

and aggregation. Lexicalisation forms the focus of Chapter 4. Referring expression generation and aggregation are left for others.

This module is so named, because in the implementation used in Chapters 4 and 5, this is the module that takes the (template, bindings) pairs and invokes the templates' text-specification functions. This template expansion process results in specifications which can then be passed to a realisation engine to generate a surface text.

3.5.5 Realisation (R)

Realisation is broadly unchanged from the consensus architecture, and in the implementation used in this work, relies heavily on the off-the-shelf SimpleNLG realisation engine from Gatt and Reiter (2009). This is the process of taking a text specification such as that shown in Figure 3.7, and turning it into text. Because the templates have already largely decided the syntax to be used, the main role this plays is in helping to ensure that the surface text is morphologically and orthographically correct.

If desired, depending on the application, the structure realisation subprocess inserts any document mark-up, such as HTML links, that may be needed. This is unchanged from the consensus architecture.

3.6 An example generation

In order to help demonstrate how the template system works, this section performs an example generation over a small provenance graph. The graph in question is shown in Figure 3.3, and shows one entity being generated from another entity by an activity associated with an agent. Figure 3.4 shows the same graph, in the Turtle notation format. In this case, one sentence is generated using one template. In practice this would involve the sentence selection process described in Section 3.3, but for now, it is helpful to focus only on the generation of a single sentence.

The first step of the generation is detecting whether or not a particular template is applicable to the content of a graph. In this case, this is evaluated using the SPARQL query in Figure 3.5, producing a single set of bindings. In a larger graph, it is possible that this query would return multiple different (template, bindings) pairs for the same template. In the process outlined in Section 3.3, the bindings functions of all the templates are executed over the graph, potentially resulting in many (template, bindings) pairs.

The second step, which is important to the process described in Section 3.3, but less important in this example, is that the coverage set for each (template, bindings) pair is calculated. The coverage set for this combination of template and bindings can be seen

in Figure 3.6. As can be seen from this set, this (template, bindings) pair can be said to cover seven of the eight triples in the example graph. In some cases, the coverage set may include triples not explicitly included in the graph being processed, but which can be inferred. In this case, the graph explicitly included triples declaring the types of the various resources, but even if this had not been the case, these triples would be included in the coverage set, as they can be inferred from the PROV relations that were in the graph.

For sentences that are eventually chosen by the processed detailed in Section 3.3, using the coverage sets calculated above, there is a third stage. In this stage, the template's text-specification function is invoked with the set of bindings as its input. In this function, the URIs are tokenised, and various NLP techniques can be applied. In this case, a verb is extracted, somewhat trivially, from the URI of the activity. Additionally, nouns are extracted for the two entities, and are determined to be singular. The details of this function's operation and the NLP used are described in greater detail in Chapter 4. From these pieces of information, the text specification shown in Figure 3.7 can be generated.

The fourth step is realisation. In our implementation, the text-specification was passed as a JSON object to a server that invoked the existing SimpleNLG realisation engine, which realised and returned the sentence "Derek illustrated chart 1 from composition.". Here it should be noted that it is the realisation engine that correctly conjugates the verb 'illustrate', and produced the correct capitalisation and punctuation. As previously stated, this is one of the great advantages of this approach, meaning that the existing research and development effort that has been placed in tools like SimpleNLG can be reused.

3.7 Summary

This chapter has described a novel template-based architecture to natural language generation. Building upon, but adjusting where necessary, the traditional approaches to explanation generation, this contribution to the state-of-the-art offers a way to transform domain-generic provenance graphs into natural language on-demand. Transformations of this nature, which were previously limited to very domain-specific applications, will potentially allow system developers to make more powerful use of their provenance data. Additionally, the introduction of the set-cover sentence selection algorithm will allow for the efficient generation of explanations from larger provenance graphs. However, there remains much research work to be done before such a fully-functional system utilising these techniques could be implemented. This work is described in more detail in the following chapter.

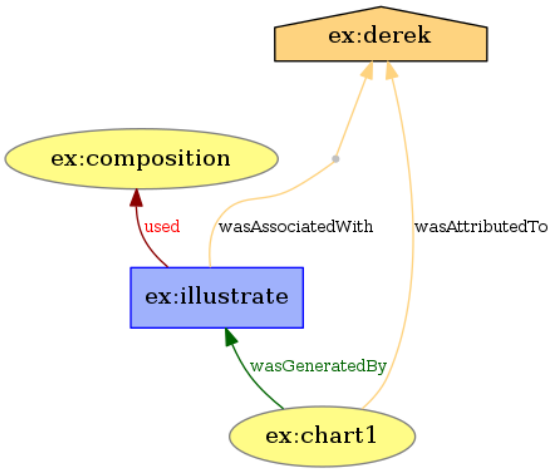


Figure 3.3: A subgraph of a provenance graph from the Southampton University Provenance Store. Full graph is available at https://provenance.ecs.soton.ac.uk/store/documents/1979/.

```
ex:illustrate prov:wasAssociatedWith ex:derek ;
prov:used ex:composition .
ex:chart1 prov:wasAttributedTo ex:derek .
ex:derek a prov:Agent .
ex:chart1 a prov:Entity .
ex:composition a prov:Entity .
ex:illustrate a prov:Activity .
ex:chart1 prov:wasGeneratedBy ex:illustrate .
```

Figure 3.4: The same subgraph as Figure 3.3, but in the Turtle notation format.

Query:

```
SELECT ?act, ?agent, ?ent1, ?ent2 WHERE {
?act prov:used ?ent2 .
?ent1 prov:wasGeneratedBy ?act .
?act prov:wasAssociatedWith ?agent .
}
```

Bindings:
{(?act, ex:illustrate),
(?ent1, ex:chart1),
(?ent2, ex:composition),
(?agent, ex:derek)}

Figure 3.5: The SPARQL query used in this template, and the bindings it generates when executed over this graph.

{(ex:illustrate, rdf:type, prov:Activity),
(ex:chart1, rdf:type, prov:Entity),
(ex:composition, rdf:type, prov:Entity),
(ex:derek, rdf:type, prov:Agent),
(ex:illustrate, prov:used, ex:composition),
(ex:chart1, prov:wasGeneratedBy, ex:illustrate),
(ex:illustrate, prov:wasAssociatedWith, ex:derek)}

Figure 3.6: The coverage set of the set of bindings extracted from the example graph using the example template.

$$
\begin{bmatrix}
\textsc{Subject} & \begin{bmatrix} \textsc{Type} & \text{noun phrase} \\ \textsc{Head} & \text{'derek'} \\ \textsc{Modifiers} & \{\} \\ \textsc{Features} & \begin{bmatrix} \textsc{number} & \text{singular} \end{bmatrix} \end{bmatrix} \\
\textsc{Verb} & \begin{bmatrix} \textsc{Type} & \text{verb phrase} \\ \textsc{Head} & \text{'illustrate'} \end{bmatrix} \\
\textsc{Object} & \begin{bmatrix} \textsc{Type} & \text{noun phrase} \\ \textsc{Head} & \text{'chart 1'} \\ \textsc{Modifiers} & \{\} \\ \textsc{Features} & \begin{bmatrix} \textsc{number} & \text{singular} \end{bmatrix} \end{bmatrix} \\
\textsc{Modifiers} & \left\{ \begin{bmatrix} \textsc{Type} & \text{preposition phrase} \\ \textsc{Noun} & \text{'composition'} \\ \textsc{Preposition} & \text{'from'} \end{bmatrix} \right\} \\
\textsc{Features} & \begin{bmatrix} \textsc{Tense} & \text{past} \end{bmatrix}
\end{bmatrix}
$$

Figure 3.7: The AVM showing the text specification generated by this template with the bindings generated from the graph in Figure 3.3. When realised, this text specification produces the text "Derek illustrated chart 1 from composition."

Chapter 4

Exploiting URIs for Informally Encoded Lexical Information

The previous chapter presented a more sophisticated architecture for generating explanations from provenance graphs, making use of state-of-the-art NLG technologies. More specifically, using a realisation engine could lead to richer sentences being generated, but to use it to the fullest extent, a greater amount of linguistic information — in particular, a wider range of verbs — is needed than would otherwise be required at runtime. The contribution of this chapter is to propose and demonstrate the effectiveness of exploiting the linguistic information informally encoded in URIs to enrich natural language explanations generated from provenance graphs. It begins by discussing some of the issues surrounding this approach, before explaining the mechanisms used to extract this linguistic information and the investigations conducted to develop them. The chapter closes with a human evaluation, examining the potential effectiveness of such an approach, demonstrating significant improvements in terms of grammatical correctness, fluency, and comprehension. *The primary contribution of this chapter is to introduce, develop, and evaluate the effectiveness of using NLP to extract the linguistic information — in particular, verbs — from the URIs denoting instances of PROV classes, and using this to enrich natural language sentences generated from that provenance.*

As discussed in Section 2.3, the related work closest to this is that of Sun (2008). Throughout this chapter, the approach to tackling the lexicalisation challenge will be contrasted and compared to that of Sun (2008), explaining the differences between the two methods, and taking into account the different goals and objectives of the two approaches.

Because Sun (2008) was attempting to solve the more general problem of converting any RDF to NL, his templates were derived from common patterns in ontology URIs, such as the fact that properties often start with a verb, whereas classes contain mostly nouns. He discovered, and defined rules for, these patterns such that he could generate sentences

about instance data based on that information. As a consequence of this method, his system would only be able to generate sentences equivalent to those sentences generated by the system presented here that *does not* exploit the linguistic information in URIs, as his sole source of verbs appears to have been property URIs in ontologies. In PROV, there are few enough of these properties that it is possible to manually encode these templates, and an ability to automatically extract this information provides no additional gain. In contrast, templates that attempt to exploit linguistic information from URIs in the way presented in this chapter, attempt to extract this information from instance data, potentially providing a far greater range of verbs.

4.1 Tokenising URIs

There are two layers of structure that exist within most URIs. The first, defined by the IETF RFCs 3986 (URIs) and 3987 (IRIs), concerns the technical information stored within a URI, and is shown in Figure 4.1. The scheme tells software which approach to use to resolve the URI, whilst the authority in most cases will be the domain name or IP address of the server hosting this information. The authority can also describe which port to connect on, though this is often omitted and instead inferred from the scheme. Together the path, query, and fragment inform the server which resource is being requested, and how they are interpreted is left to the server to decide.

Whilst this structure will be useful to the approach to lexicalisation presented here, of greater concern is the second layer of structure — the linguistic layer. In this case, rather than trying to split the URI into its various technical components, the goal is to split it into its linguistic components — most commonly words and numbers. These components are called *tokens*, and the process of splitting a URI into these tokens *tokenisation*. Because of their technical structure, the parts of the URI that are most likely to contain relevant linguistic information are the path, the query, and the fragment. Consequently, these are the only parts of the URIs analysed here.

Tokenisation is important from a computational linguistics perspective because most existing NLP techniques rely on a text having been tokenised before they are applied. For example, the part-of-speech tagging described later in Section 4.2, takes as its input an ordered list of tokens. By taking a URI and being able to reduce it into a list of tokens, it is possible that the same techniques could also be applied to URIs. This means that the consequence of not being able to accurately tokenise a URI is that it results in noise being fed to the later processes in the NLP pipeline. As tokenisation is the first stage of the NLP pipeline, bad tokenisation would consequently affect every part of the system, and in the case of the provenance explanation generation described in this chapter, could eventually result in bad, unclear, and potentially ungrammatical

explanations being generated. The study in Section 4.4 attempts to quantify the impact this might have on the generated sentences.

Here, it is important to pause and consider the fact that URIs are, per the specifications, considered to be opaque. This means that it is considered improper (Berners-Lee, 1996) to use the internals of a URI for anything other than identification and resolution. However, in this investigation, with over 95% of the URIs in the ProvStore URI dataset found to contain linguistic information of one kind or another. To be clear, this linguistic information was not always a fully linguistic URI, such as some of the examples given below, but was often of the form /[class_name]/[UUID], which at least provides information as to what type of object a URI denotes. The remaining URIs had paths consisting entirely of numbers, symbols, or acronyms and abbreviations. As an example from the ProvStore URI dataset, `http://ec.europa.eu/food/safety/rasff#fsa_first_statement` clearly denotes a statement — the first statement — possibly made by an organisation called the FSA or about an organisation called the FSA, about food safety — even for a human reading the URI there is a degree of ambiguity as to what it denotes, though it is clearly more meaningful than a simple numeric identifier. A natural implication of this ambiguity is that it is impossible to guarantee that any explanation generated using the linguistic information in URIs will be factually correct. However, given that more sophisticated NLG techniques require a greater degree of linguistic information at the realisation stage — in particular, a wider variety of verbs — it is the contention of this work that utilising the linguistic information within URIs is, nevertheless, of key importance in tackling the challenge of generating more natural explanations from PROV graphs. One of the aims of this chapter is to demonstrate this.

Nevertheless, developers and those minting URIs are under no obligation to make them meaningful, and it would be as valid for the URI in Figure 4.1 to refer to a car manufacturer specification sheet, for example, as it would a cake recipe. In practice however, it would appear unlikely that developers would want to use URIs in such a misleading manner — more likely, perhaps, is the possibility that as systems develop over time, gradually the usage of certain URIs might change so that the linguistic meaning informally encoded in the URI no longer matches exactly with the concept that the URI denotes. A white paper by the Open Data Institute and Thomson Reuters (2014) recognised the problem of unstable identifiers and proposed a number of ways to help make identifiers longer-lasting and more meaningful, however many datasets fail to meet this standard. On the other hand, as the alternative to extracting the linguistic information from URIs typically involves presenting the user with a raw URI, which would be just as misleading to the user, both of these approaches suffer from this problem, which could only be remedied by never showing the user a URI or anything derived from one.

In addition, unlike the technical structure of URIs, the demarcation of the linguistic tokens within URIs is unspecified, and consequently tends to vary from one data source to another (see Table 4.1). Nevertheless, the technical structure of a URI often does have

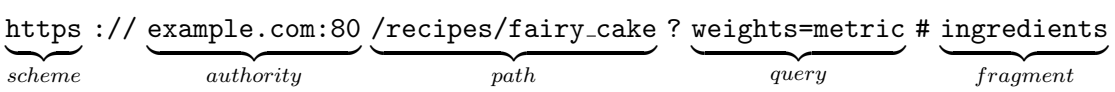


Figure 4.1: The technical structure of a URI — Only the scheme and the path are mandatory, though the path may be empty.

URI Segment	Tokens
`The_Lord_Of_The_Rings`	The, Lord, Of, The, Rings
`JRRTolkien`	JRR, Tolkien
`J._R._R._Tolkien`	J, R, R, Tolkien
`BritishFantasyNovels`	British, Fantasy, Novels
`chapters/11`	chapters, 11

Table 4.1: Examples of tokenised URI segments

a bearing on the linguistic structure: firstly, it does not make sense for linguistic tokens to span more than one technical segment of a URI; and secondly, the linguistic information specific to the resource in question is most likely to be found in the path, query, and the fragment, with the more specific information tending to be further towards the right end of the URI. Where multiple tokens appear in the same technical segment, they are in practice often demarcated by the use of camel case or snake case. For example, `http://iri.nidash.org/cluster_definition_criteria_id` uses snake case, whereas `http://clingen.org/Groups/PhenoGroup1` uses camel case. It should be noted that there is not always a single correct tokenisation, as seen with J. R. R. Tolkien in Table 4.1. In such cases, either tokenisation can be considered acceptable.

It is possible to approach this problem from a number of directions, with the two most immediately obvious being to build a system with an understanding of the language sufficient to be able to recognise tokens in that language, or instead to detect patterns often found in the symbols used to represent the linguistic tokens, such as the use of capitalisation and punctuation. Sun (2008) does not describe which, if either, approach to tokenisation he used.

The linguistic-based approach would require the program not only to recognise terms in a known lexicon but also to be able to recognise previously unknown words such as proper nouns and specialist terms. The former challenge, recognising known terms, is difficult because many terms in the lexicon can be conjugated, pluralised, or abbreviated, significantly increasing the total number of tokens to possibly find, but also because many linguistic tokens share common substrings with — or are substrings of — other tokens and would need to be disambiguated. The latter challenge, identifying tokens as previously unknown words, requires a large degree of intuition, and is consequently a much harder challenge to solve programmatically.

Alternatively, the pattern-based approach also has a number of subtleties that could confuse a computer. For instance, during the investigation in Section 4.1.1, a number of

edge cases were found where the correct solution was trivially obvious to the investigator, but required a significant extension to the regular expression. For example, Scottish and Irish surnames, among others, often contain a capital letter in the middle of a word, confusing the regular expression into thinking there were two tokens in camel case — for example, `http://www.ipaw.info/data/people/McGuinnessDeborahL`.

As for which approach a human might use, this is an issue more for cognitive psychology than computer science. Nevertheless, it is perhaps worth observing that Ancient Greek and other ancient Mediterranean languages were commonly written with neither letter-case nor spaces between words — a style called *scriptura continua* (Saenger, 1997). Consequently, it seems reasonable to suggest that humans can do this tokenisation linguistically — though there is no guarantee than modern humans would be able to do it without training. Saenger (1997) posits that reading aloud is an important part of the cognitive process of parsing such a text, with spaces being introduced at approximately the same time as larger numbers of people were learning to read silently.

4.1.1 A regular expression for tokenising URIs

For this investigation, as well as all the investigations in this chapter, the source data came from the University of Southampton Provenance Store (ProvStore)[1], and was used in accordance with University of Southampton ethics approval ERGO/FPSE/16722. The 63,335 documents in the ProvStore that had been made publicly-available by their creators were downloaded, and for each instance of a PROV class in those graphs (i.e. a prov:Entity, prov:Activity, prov:Agent, prov:Derivation, prov:Generation, etc.), the URI denoting that resource and the class of that resource were stored in a database. Because some of the provenance graphs were very large and repetitive, with very similar URIs, similar graphs and duplicate URIs were removed from the dataset, leaving 2637 for the investigations. This will henceforth be referred to as the ProvStore URI dataset.

Wanting to derive a regular expression to tokenise the URIs denoting PROV resources, our investigation proceeded iteratively, starting with a simple regular expression that simply split on non-alphanumeric characters. This regular expression was then applied to all the URIs in the ProvStore URI dataset. One by one, each URI in the dataset was marked as correctly tokenised, and the correct tokens stored, until an incorrectly tokenised URI was discovered. Each time this occurred, the regular expression was adapted to be able to tokenise it correctly, or the URI was marked as untokenisable. The new regular expression was then applied to all of the URIs in the dataset, automatically checking the tokenised URIs previously marked as correct for a regression in behaviour.

This process was continued until all of the URIs were either marked as correctly tokenised or marked as untokenisable. This resulted in over 95% of the URIs being tokenised

[1] `https://provenance.ecs.soton.ac.uk/store`

```
[0-9a-fA-F]{10,}|
(?:Mc|Mac)?[A-Z][a-z]+|
[A-Z]+s?(?![a-z])|
[a-z]+|
[0-9]+
```

Figure 4.2: The tokenisation regular expression. Note that the line breaks were added for the purposes of printing, and do not appear in the actual regular expression.

correctly with the regular expression shown in Figure 4.2. The remaining 5% were unable to be tokenised because they required a greater degree of linguistic knowledge to be tokenised correctly, or the investigator was unable to tokenise the URI themselves. For example, with the URI `http://data.usewod.org/dataset/linkedgeodata`, the regular expression is unable to split the final part of the path "linkedgeodata" correctly into "linked" and "geodata" because there was no capitalisation pattern to notice, and a lexicon would most likely have been required to have spotted the two terms. It is unknown how representative the ProvStore URI dataset is of PROV graphs in general, so it is potentially incorrect to say that this regular expression is capable of tokenising over 95% of all URIs denoting PROV resources, but this result is nevertheless encouraging.

The regular expression comprises five major components, which can be seen in Figure 4.2, and where the order of the components is significant:

1. `[0-9a-fA-F]{10,}` — Detects hexadecimal numbers of 10 characters or more. This lower limit was introduced, to avoid splitting English words like 'feedback'. There were no hexadecimal numbers shorter than 10 characters in the dataset, as most appeared to be UUIDs.

2. `(?:Mc|Mac)?[A-Z][a-z]+` — Detects words beginning with a capital letter. This is primarily used to split strings in camel case. One exception that had to be added to this part of the regular expression was to account for Scottish surnames — a number of which appeared in the dataset — and which have a capital letter in the middle of a token.

3. `[A-Z]+s?(?![a-z])` — Detects acronyms and their plurals, provided that their plurals are signified by an 's', which is not followed by another lower-case character.

4. `[a-z]+` — Detects lower-case strings.

5. `[0-9]+` — Detects decimal numbers.

4.2 Tagging URIs

Having tokenised the URIs with an acceptable level of accuracy, the next step was to develop a way of understanding what roles each token in the URIs played linguistically. These roles, called *parts of speech*, refer to the different types of word that can appear in language — for example nouns, verbs, and adjectives. In order to reduce the amount of development time required, it was desirable to find out how well an off-the-shelf part-of-speech (POS) tagger would perform at classifying the various tokens within a URI. It was unclear if this would work particularly well, as URIs tend to be very ungrammatical, whereas the off-the-shelf taggers have all been trained on longer, presumably grammatically correct texts.

As there is no publicly available corpus of tokenised and POS-tagged URIs, it was necessary to build one to test the classifier against. Using the ProvStore URI dataset, with now-tokenised URIs, all 2637 URIs (15169 tokens) were manually tagged by the investigator. Following this, the off-the-shelf POS-tagger — a maximum entropy tagger (Ratnaparkhi, 1996) trained on the Penn Treebank (Marcus et al., 1993), which was the default tagger in the Python NLTK library at the time (Bird et al., 2009) — was used to also tag all of the tokens in the dataset. The set of tags used for this are explained in Appendix A. The effectiveness of this automated tagging is shown in Table 4.2 and the two confusion matrices in Figures 4.3 and 4.4. The second confusion matrix shows the tags grouped into larger, less-granular categories. This grouping was performed because when exploiting this linguistic information for generation purposes, it is less important to know exactly what type of verb — for example — a word is, but rather that it is simply a verb. The mapping of University of Pennsylvania tags to these broader tag groups is shown in Table 4.3. The groups *Noun*, *Verb*, and *Number* are self explanatory; *Modifier* is comprised of adjectives and adverbs (they modify the verb or noun to which they are attached); *Other* is comprised of the remaining tags that do not fit into the other four groups.

The results of this classification are shown in Table 4.2. It uses a number of classification metrics, which can be used to assess the performance of a classifier, such as a POS-tagger. The first four columns — TP, FP, TN, FN — show the raw number of occurrences of true positives, false positives, true negatives, and false negatives respectively. In this case, *true* refers to a correct classification, and *false* an incorrect classification. *Positive* refers to an instance that the classifier identified as belonging to the class, and *negative* an instance the classifier identified as not belonging to the class. Accuracy refers to the proportion of instances classified correctly, i.e. $Accuracy = \frac{TP+TN}{TP+FP+TN+FN}$. Precision is the proportion of instances identified as belonging to the class that actually did belong to the class, i.e. $Precision = \frac{TP}{TP+FP}$. Recall is the proportion of instances belonging to a class that were identified as such, i.e. $Recall = \frac{TP}{TP+FN}$. The F1 score is the harmonic mean of the precision and the recall, i.e. $F1 = \frac{2 \times Recall \times Precision}{Recall+Precision}$.

At first glance, these results look very promising, with an accuracy rate of 92.3%, when the largest group — nouns — only comprises 81.9% of the tokens in the dataset (see Table 4.4). Consequently, the off-the-shelf tagger performs 12.7% (or 10.4 percentage points) better than the most trivial possible classifier, which would be to classify all tokens as nouns. However, it is not particularly difficult to tell the difference between words and numbers represented as digits. A classifier that classified all tokens containing digits as numbers, and everything else as a noun would have an accuracy of approximately 94.4%[2], which is 2.3% (or 2.1 percentage points) better than the performance of the off-the-shelf tagger. Nevertheless, in order to create richer sentences, it is particularly necessary to be able to extract verbs and modifiers from URIs. The off-the-shelf tagger was able to correctly classify 55% and 43% respectively of those two types of tokens correctly (see Recall in Table 4.2). As the motivation for using a part-of-speech tagger is to be able to extract this richer linguistic information, being able to detect 55% of verbs and 43% of modifiers is considerably better than nothing, which is what would be achieved without such a classifier.

As can be seen from the confusion matrix in Figure 4.3, and to a lesser extent Figure 4.4, the POS-tagger is repeatedly overclassifying tokens as Nouns, across many classes (resulting in the light-grey vertical line). As with many machine learning approaches, it is difficult to see exactly why this is happening, though it would not be unreasonable to speculate that this is a result of the different linguistic structures between natural English and URIs — it is possible that some of the features that are omitted when constructing a URI or URI schema are precisely those that are useful for determining the true class of a token, with the tagger defaulting to Nouns when the necessary information is not present. Nevertheless, the diagonal line is much clearer in both confusion matrices, highlighting that overall, the POS-tagger is classifying well, as supported by the numerical analysis above.

Class	TP	FP	TN	FN	Accuracy	Precision	Recall	F1
Noun	11673	340	2400	756	0.93	0.97	0.94	0.96
Verb	233	622	14121	193	0.95	0.27	0.55	0.36
Modifier	129	118	14750	172	0.98	0.52	0.43	0.47
Number	1883	6	13262	18	1.00	1.00	0.99	0.99
Other	79	86	14971	33	0.99	0.48	0.71	0.57
Overall	13997	—	—	1172	0.92	—	—	—

Table 4.2: The performance of the off-the-shelf POS-tagger when tagging tokenised URIs from the ProvStore URI dataset, grouped by tag type. TP = true positive; FP = false positive; TN = true negative; FN = false negative.

[2]This figure is approximate because a small minority of the tokens classified as numbers in the dataset were expressed as words, e.g. *forty*, *two*.

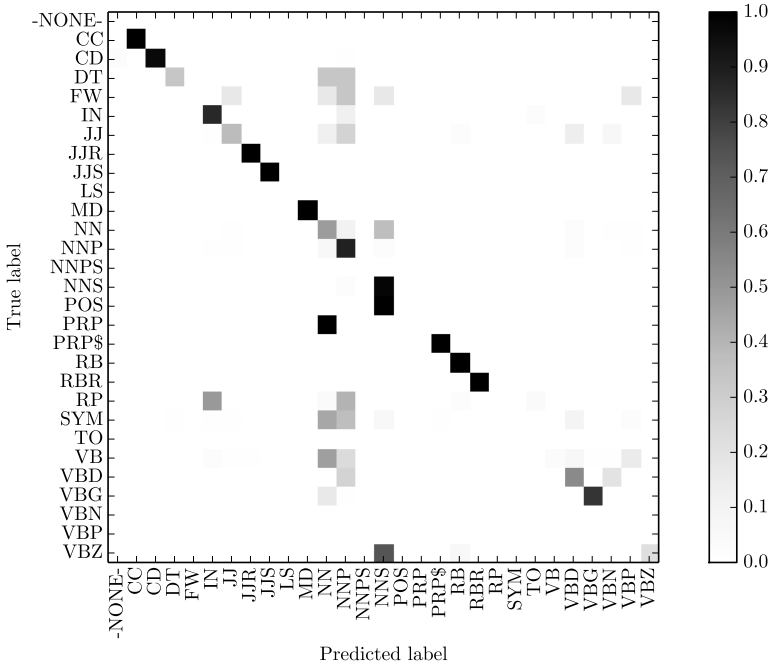


Figure 4.3: Confusion matrix showing tags as predicted by the POS-tagger and the true tag for each token; normalised by number of true occurences of each tag. Accordingly, each row sums to 1.0, except rows where the tag did not truly exist in the dataset, which are left blank). The labels are the tags explained in Appendix A

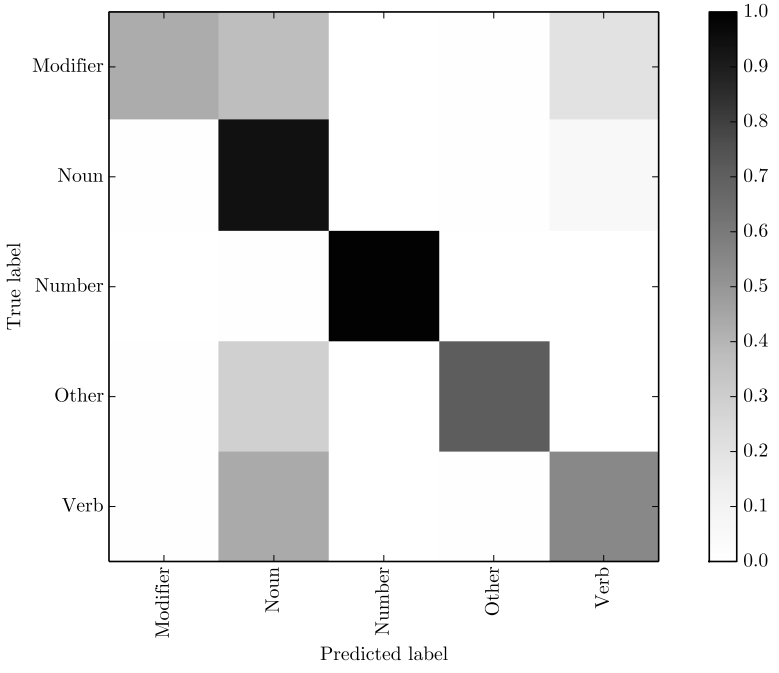


Figure 4.4: Confusion matrix showing tag groups as predicted by the POS-tagger and the true tag group for each token; normalised by number of true occurences of each tag group. Accordingly, each row sums to 1.0.

Group	**Tags**
Noun	FW, NN, NNP, NNPS, NNS, SYM
Verb	MD, VB, VBD, VBG, VBN, VBP, VBZ
Modifier	JJ, JJR, JJS, RB, RBR
Number	-NONE-, CD
Other	CC, DT, IN, LS, POS, PRP, PRP$, RP, TO

Table 4.3: Mappings of University of Pennsylvania tags to the tag groups used in this work. See Appendix A for descriptions and examples of each tag.

Class	%
Nouns	81.9%
Verbs	2.8%
Modifiers	2.0%
Numbers	12.5%
Other	0.7%

Table 4.4: The composition of the ProvStore URI dataset by tag group.

4.3 Building more advanced templates

The purpose of extracting this extra linguistic information is to make it possible to create templates capable of generating richer sentences. Without it, the templates are limited to using the verbs built in to the PROV model, such as "was derived from" or "was associated with". As an example, the graph shown in Figure 4.6 can be described by the sentence "ex:cake was generated by ex:baking, which was associated with ex:John." without needing to extract any linguistic information, leaving the user to interpret the URIs. Alternatively, having extracted the linguistic information, it would be possible to generate a sentence like "John baked cake.". The bindings query (shown in Figure 4.7) remains the same, but the text specification generated by the template is different (see Figure 4.8), where the verb phrase is able to use 'baking' as its head, which is then correctly conjugated by the realisation engine as 'baked'.

In order to make use of the information informally encoded in the URIs denoting instances of PROV classes, the pipeline shown in Figure 4.5 is used: firstly, the URIs are tokenised, using the regular expression; then they are then tagged using an off-the-shelf POS tagger. Thirdly, logic uses the tags to determine whether the URI contains useful information, such as verbs, nouns, whether any nouns are plural, etc. This step is important, because some of the templates will only work if those features are present, and other templates require that information to pass on to the realisation engine. Finally, this information is used by the templates to decide whether or not that URI provides enough information for that template to be used — in the case that it does not, a different fall-back template is used that does not attempt to exploit this information.

To be clear about the processes shown in Figure 4.5, the tokenisation and POS-tagging stages are the same as those described earlier in Sections 4.1.1 and 4.2. In the implementation used in the process of the experiments in this chapter and the next, the process called "Logic" is part of the implementation of the templates' bindings function and text-specification function. Because these functions are implemented in Python, the developer of a template can introduce any arbitrary logic into their templates. However, in the standard templates used for the experiments in this chapter and Chapter 5, there are two primary uses for this logic: filtering and providing linguistic information to the realisation engine.

Filtering In addition to the SPARQL query used by the bindings function to match certain patterns in the graph, there is logic in this function that can be used to potentially filter out a number of those sets of bindings. This is important, for instance, in the templates that require a verb to have been extracted in order to be useful, such as in Figure 4.7 — if the URI ex:baking had not contained a verb, or the tagger could not identify one, then it would not have been appropriate to

have used this template. If that had been the case, that particular set of bindings could have been removed from the set of sets returned by the bindings function.

Aiding the realiser Additional logic can also be invoked in the templates' text-specification functions, where, in particular, it is important to know whether nouns are singular or plural. As can be seen in the example text specification shown in Figure 4.8, such features appear multiple times, and are subsequently used by the realisation engine to determine how to, for example, accurately conjugate verbs. Consequently, the implementation of the system used in this work differs slightly from this pipeline as shown, in integrating the logic into the templates, but conceptually it performs the same role.

There is, however, no limitation on the logic that can be used within these templates. For example, if a developer wanted a template to be rendered differently in the run-up to Christmas, they could implement these changes in Python, as part of the logic of the templates, using Python's built-in `datetime` library to identify when Christmas was coming. This feature adds a lot of flexibility to the sorts of templates that can be created for this architecture. However, perhaps most importantly, it allows for templates to be created that can handle PROV graphs with differing level of extractable linguistic information in URIs.

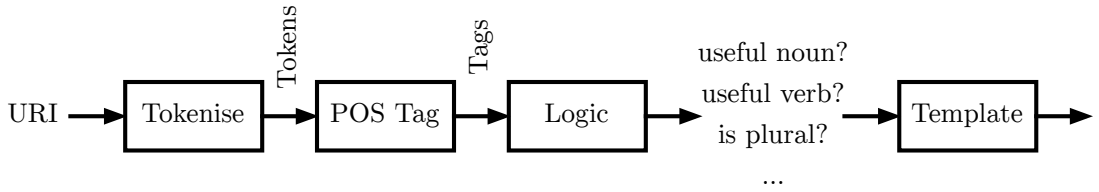


Figure 4.5: The pipeline showing how a URI is processed to make a decision as to whether it contains linguistic information that can be exploited.

4.4 Human evaluation

Having created templates, described in Chapter 3, able to exploit the linguistic information in URIs, extracted using the techniques described in the preceding sections, the next step was to evaluate how sentences generated using those templates compared to those generated by templates that do not exploit linguistic information in URIs. In order to do this, a human evaluation was devised that would allow for direct comparison of pairs of sentences. In each pair of sentences, one was generated taking advantage of the additional linguistic information extracted from URIs, and the other was generated without this information.

In terms of selecting the sentences to use, six graphs were arbitrarily chosen from the provenance store, created by five different authors, and consequently representing

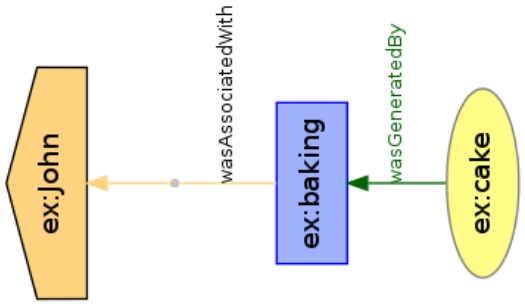


Figure 4.6: An example provenance graph

```
SELECT ?entity ?activity ?agent WHERE {
   GRAPH <prov_graph> {
      ?entity a prov:Entity .
      ?entity prov:wasGeneratedBy ?activity .
      ?activity a prov:Activity .
      OPTIONAL { ?activity prov:wasAssociatedWith ?agent} .
      OPTIONAL { ?entity prov:wasAttributedTo ?agent} .
      FILTER bound(?agent)
    }
 }}
```

Figure 4.7: An example bindings query, which looks for generations in the provenance graph where the generating activity was associated with, or the generated entity was attributed to an agent.

$$\begin{bmatrix} \textsc{Subject} & \begin{bmatrix} \textsc{Type} & \text{noun phrase} \\ \textsc{Head} & \text{'John'} \\ \textsc{Modifiers} & \{\} \\ \textsc{Features} & \begin{bmatrix} \textsc{number} & \text{singular} \end{bmatrix} \end{bmatrix} \\ \textsc{Verb} & \begin{bmatrix} \textsc{Type} & \text{verb phrase} \\ \textsc{Head} & \text{'baking'} \end{bmatrix} \\ \textsc{Object} & \begin{bmatrix} \textsc{Type} & \text{noun phrase} \\ \textsc{Head} & \text{'cake'} \\ \textsc{Modifiers} & \{\} \\ \textsc{Features} & \begin{bmatrix} \textsc{number} & \text{singular} \end{bmatrix} \end{bmatrix} \\ \textsc{Modifiers} & \{\} \\ \textsc{Features} & \begin{bmatrix} \textsc{Tense} & \text{past} \end{bmatrix} \end{bmatrix}$$

Figure 4.8: An AVM showing an example text specification for a template involving an activity generating an entity, where the activity was associated with an agent. When realised, produces the sentence "John baked cake."

five different domains. Explanations were then generated from these graphs using the PROVglish architecture described in 3, using both the templates that exploited the linguistic information in URIs, as well as those that did not. From these explanations 15 matching pairs of sentences were chosen at random. Each of the five graphs was represented by at least one sentence in the test set, and at most five. Of the sentences in the test set, nine were from graphs that appeared to have been generated by an automated process (four of the graphs), and six appeared to be from graphs that had been created by hand (two of the graphs). The characteristics of the graphs are summarised in Table 4.5.

Graph	Nodes	Edges	Entities	Activities	Agents
1	24	73	12	4	2
2	14	18	6	1	1
3	1756	5154	1040	34	11
4	163	415	21	12	8
5	42	61	10	5	2
6	13	16	1	2	2

Table 4.5: Metrics showing the characteristics of the six graphs used in this experiment. Note: the nodes and edges values are calculated based on the PROV-O RDF graph, which uses a combination of qualified and simple PROV relationships.

Three dimensions were chosen, related to those used in Lester and Porter (1997): grammatical correctness, fluency, and comprehensibility. Whilst it is more often conventional to have experts perform these evaluations as in Lester and Porter (1997) and Sun (2008), because the primary concern of this work is how these sentences will perform when presented to non-expert users of provenance, this evaluation was conducted without specifically seeking experts on provenance. Similarly, the terms 'grammatical correctness', 'fluency', and 'comprehensibility' were not defined for the participants, in the belief that they are common enough terms to have been understood by all of the participants. In practice, the participants were drawn from the University of Southampton community, with the majority from the Electronics and Computer Science academic unit. As a consequence of this fact, a number of the participants were familiar with provenance generally, and PROV more specifically, though these participants were in the minority.

All participants were selected for their ability to speak English, with a general requirement that all participants be native English speakers, or have lived in the United Kingdom for sufficient time to have become fluent. This requirement, more than anything, motivated that this evaluation should be conducted in person, rather than using an automated online crowd-sourcing approach such as using Amazon's *Mechanical Turk*[3], so as to be able to ensure this level of control. Participants were incentivised to take part in this study by being entered into a prize draw for one of two £50 vouchers. The

[3] `https://www.mturk.com`

study was conducted in accordance with University of Southampton ethics approval ERGO/FPSE/16731.

The study was conducted with each participant separately, using a locally-hosted Web application, with the interface shown in Figure 4.9. Each participant was shown all of the sentence pairs, one at a time, in a randomised order. For each sentence pair, one sentence was called *Sentence A* and the other *Sentence B*, with them shown side-by-side on the page — which sentence was which was randomised each time the page was loaded. Beneath those sentences, were the questions. For each sentence pair, each participant was asked the following questions:

Q1 Is there any information in Sentence A that isn't in Sentence B? (Yes / No)

Q2 Is there any information in Sentence B that isn't in Sentence A? (Yes / No)

Q3 Do the sentences disagree at all? (Yes / No)

Q4 Do the sentences agree entirely? (Yes / No)

Q5 How would you rate Sentence A, in terms of:

- **Q5a** the grammatical correctness of the English? (1 *Very Bad* – 6 *Very Good*)
- **Q5b** the fluency of the sentence? (1 *Very Bad* – 6 *Very Good*)
- **Q5c** how easily you can understand it? (1 *Very Bad* – 6 *Very Good*)

Q6 How would you rate Sentence B, in terms of:

- **Q6a** the grammatical correctness of the English? (1 *Very Bad* – 6 *Very Good*)
- **Q6b** the fluency of the sentence? (1 *Very Bad* – 6 *Very Good*)
- **Q6c** how easily you can understand it? (1 *Very Bad* – 6 *Very Good*)

Q7 Which sentence do you think is the better explanation? (Sentence A / Sentence B / Neither)

In total, fifteen sentence pairs were shown to each of the fifteen participants, resulting in a total of 225 independent observations for each dimension.

For questions 5 and 6, in order to identify whether there was a significant difference between the performance of the two sets of templates a statistical measure had to be chosen. Conventionally, a measure like the Student's t-test (Student, 1908) might be used to establish if two samples are statistically likely to derive from the same population. However, its use would be inappropriate in this instance for a number of reasons: firstly, this t-test can only be used when the samples in question both conform to a gaussian distribution of equal variance (though Welch's t-test can be used with samples of unequal variance), and Figures 4.10, 4.12, and 4.14 show this very clearly not to be the case;

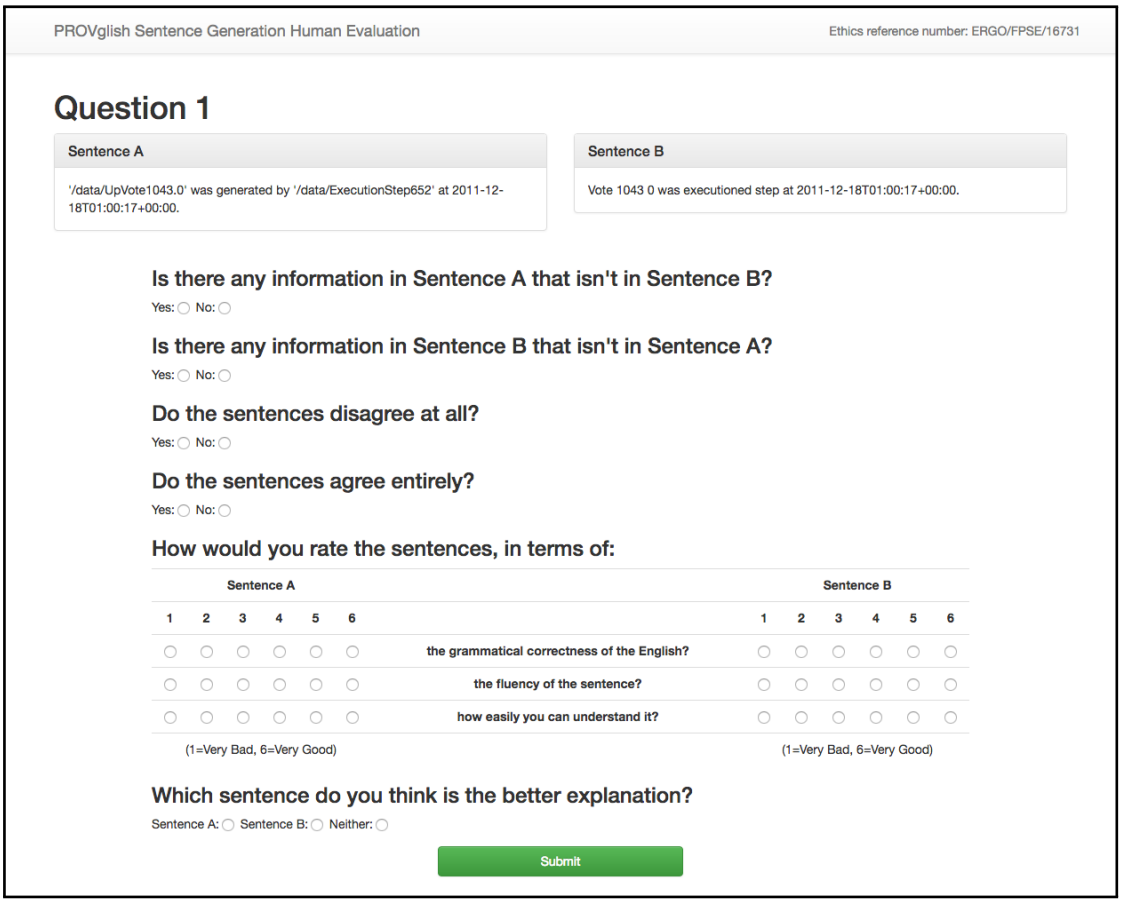


Figure 4.9: A screenshot of the user evaluation Web-interface.

secondly, and perhaps more importantly, it is not possible to use either t-test on ordinal data as concepts such as mean and standard deviation may not be applied to such data. Because it is impossible to tell whether participants would treat the 1 to 6 scales as linear or not, it is only possible to treat these responses as ordinal data, i.e. it is possible to tell that 2 is better than 1, but not necessarily by how much.

Instead, the Wilcoxon test has been chosen (Wilcoxon, 1945). Rather than working with the absolute values of the observations, this paired-value test calculates the difference between the two values in each paired observation, and then attempts to determine if these differences are symmetric around zero using weighted rankings. As a consequence of this ranked approach, it can be applied to ordinal data, such as those collected in this experiment. When using this test it is conventional to report the T value and the p value, as shown in Figures 4.11, 4.13, and 4.15. In this case, the p value represents the probability that the differences, or greater differences, between the samples would have been observed if they were indeed drawn from the same population, while the T value is the lesser of two rank-weighted sums calculated by the test, where a lower value compared to N_r implies a greater difference between the two samples. (N_r is the number of paired observations where the difference between the two values in the pair is non-zero.) T will always be between 0 and half the triangle number of N_r. Due to

this dependance on N_r it is possible to have two instances of this test where each has the same sample size (N), with one producing a greater T value than then other, yet with a lower p value. For an example, compare Figures 4.13 and 5.14, where Figure 5.14 has a higher p value despite a lower T value; a result of the fact that the value of N_r is much smaller for that sample because of the large number of 0-difference pairs as well as a smaller sample size (N).

4.4.1 Expected results

Expectation 1 The sentences generated by exploiting the linguistic information in URIs should be reported as the better explanation — ideally in all cases, but at least in the majority of cases. Similarly, if too large a percentage of responses report that neither sentence is the better explanation, that would indicate that this approach to enriching provenance explanations was not particularly effective. This will be assessed by **Q7**.

Expectation 2 In terms of grammatical correctness, the sentences generated *without* exploiting the linguistic information in URIs are known to be grammatically correct. On the other hand, the sentences that do exploit this information often have small — and occasionally large — grammatical flaws, due to imperfections in the exploitation technique and the realisation engine. Consequently, the target was for both sets of sentences to perceived as equally correct grammatically, though the expectation was that the sentences exploiting the linguistic information might perform worse. This will be assessed by **Q5a** and **Q6a**.

Expectation 3 On the other hand, the very motivation for exploiting this additional linguistic information was to create more fluent, easier to comprehend explanations. Consequently, the desired outcome was for there to be a signifiant improvement over not exploiting the linguistic information in both of these dimensions. This will be assessed by **Q5b**, **Q6b**, **Q5c** and **Q6c**.

Expectation 4 There should not be any great difference in the information perceived to be contained within the sentences generated exploiting the linguistic information in URIs and the sentences generated without exploiting that information. That is, the sentences should convey the same information. This will be assessed by **Q1** and **Q2**.

Expectation 5 As a corollary of Expectation 4, there should be no disagreement between the two sentences, and the sentences should agree completely. This will be assessed by **Q3** and **Q4**.

4.4.2 Results obtained

Result 1 When asked directly, the participants stated 56.4% of the time that they thought the sentence exploiting the information in URIs was the better explanation; 29.3% of the time they rather thought that the other sentence was the better explanation; and 14.2% of the time neither sentence was considered the better explanation. This can be seen in Figure 4.16, and supports **Expectation 1**.

Result 2 The sentences generated by exploiting the linguistic information in URIs were rated significantly higher than those generated without exploiting that information, across all three dimensions of grammatical correctness, fluency, and comprehensibility, with $p < 0.01$. Because of the relatively small sample size, it is impossible to generalise the results to the whole English speaking world, but it is possible to say with confidence that the 15 participants thought that the sentences with URIs exploited performed better than those without. These results can all be seen in the six figures 4.10, 4.11, 4.12, 4.13, 4.14, and 4.15. This contrasts with **Expectation 2** in the sense that the sentences generated without exploiting the linguistic information in URIs were previously considered to be grammatically correct, whereas the actual results show that this is not perceived to be be the case by the users. However, these results support **Expectation 3**.

Result 3 The median response in all dimensions for sentences exploiting the linguistic information in URIs was 5, in contrast to the median response of 4 for the sentences not exploiting that information. The median difference in score was 0, +1, and +1, for grammatical correctness, fluency, and comprehensibility respectively. These latter statistics show that not only did the sentences exploiting the linguistic information perform better in aggregate, but in terms of fluency and comprehensibility, they performed better in a majority of cases. This supports **Expectation 2** and **Expectation 3**.

Result 4 The mode response for each of the dimensions — grammatical correctness, fluency, and comprehensibility — was 6, for sentences exploiting the linguistic information in URIs. In contrast, the mode responses were 4, 4, and 5 for grammatical correctness, fluency, and comprehensibility respectively for the sentences not exploiting the linguistic information in URIs. This supports **Expectation 2** and **Expectation 3**.

Result 5 For each sentence pair, each participant was asked whether the sentences agreed and whether the sentences disagreed separately. The responses to these questions were a little perplexing, with participants reporting that sentences agreed 57.8% of the time and disagreed only 20.9% of the time — this can be seen in Figure 4.17. Further analysis of the data showed that 2.7% of the time the participant reported that the sentences both agreed and disagreed, and 24.0% of the time that the sentences neither agreed nor disagreed. This result perhaps derives from

the fact that users were themselves considering the information contained in the raw URIs shown in the sentences that did not exploit that information, which in many cases was not all extracted and put into the sentences that did attempt to exploit the linguistic information in URIs. This supports neither **Expectation 4** nor **Expectation 5**.

In summary, the results fully support **Expectations 1** and **3**. Some of the results support **Expectation 2**, whilst the primary statistical analysis — the Wilcoxon test — does not. However, this is not necessarily a 'bad' result, as it shows the sentences exploiting the linguistic information in URIs performing better than expected. It does however highlight that the participants may have had a different understanding of what grammatical correctness is than the experimenter — again not necessarily a 'bad' thing as the object of the experiment was to establish how ordinary users would consider the sentences, rather than how they would be considered by a computational linguist. **Expectations 4** and **5** were also not supported, and potential reasons for this have already been given.

In aggregate, however, the results show that this new approach to generating explanations from provenance graphs performs significantly better than existing domain-independent techniques, justifying further development and research. Tables 4.6 and 4.7 show the results of the study broken down by sentence pair and participant respectively.

4.5 Discussion of notable generations

In the study presented in the previous section, a number of sentences were generated by exploiting the linguistic information encoded in the URIs denoting the resources. The complete set of these sentences can be seen in Appendix B. However, in this section, a number of the more interesting features of these sentences are explored to help understand where the techniques are successful, and where further improvements could be made.

4.5.1 Extracting verbs

A number of the sentences are richer than their less-sophisticated counterparts in large part because of the fact that they are able to use verbs that have been extracted from the URIs. For example, in Sentence 14 `Derek illustrated chart 1 from composition.` is more natural than Derek deriving the chart from composition by `/illustrate`, which is the unconjugated form in the URI. The realisation engine, in this case, was able to correctly conjugate the verb.

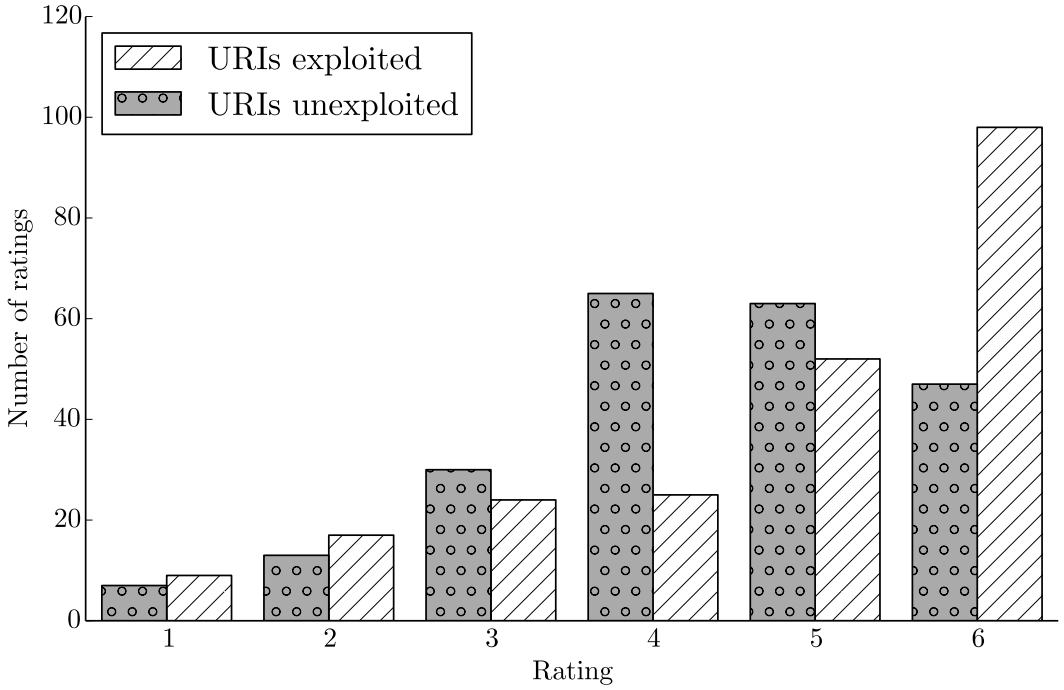


Figure 4.10: Responses to **Q5a/Q6a** "How would you rate the sentences, in terms of the **grammatical correctness** of the English?" The figure shows the absolute values of all responses by all participants to all sentence pairs.

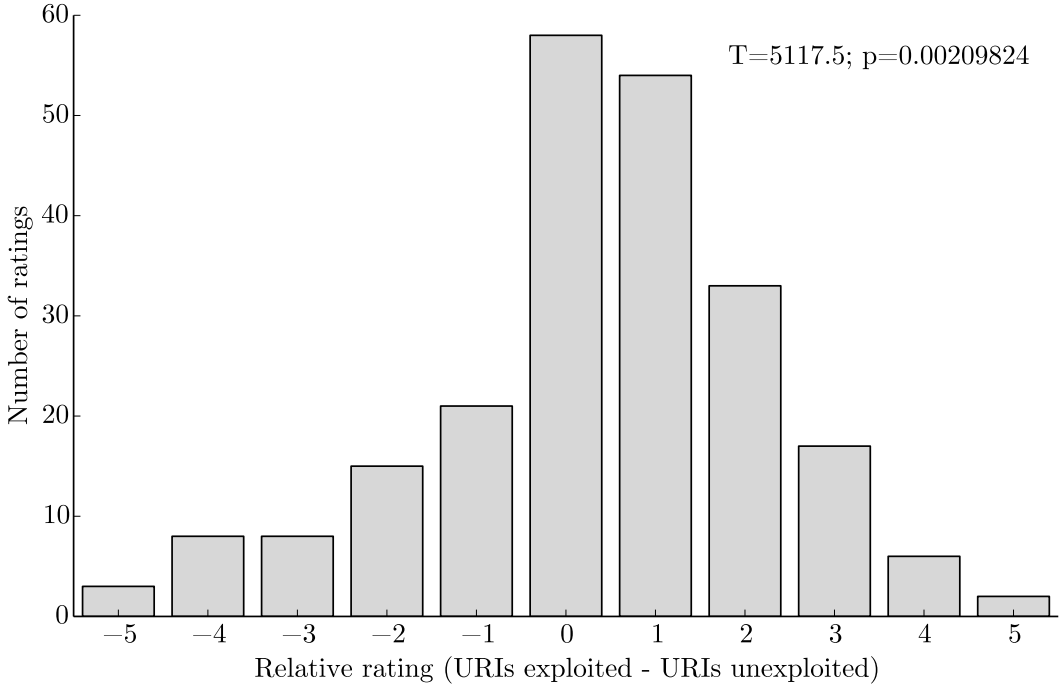


Figure 4.11: Responses to **Q5a/Q6a** "How would you rate the sentences, in terms of the **grammatical correctness** of the English?" The figure shows the relative values of all responses by all participants to all sentence pairs, (URIs exploited – URIs unexploited).

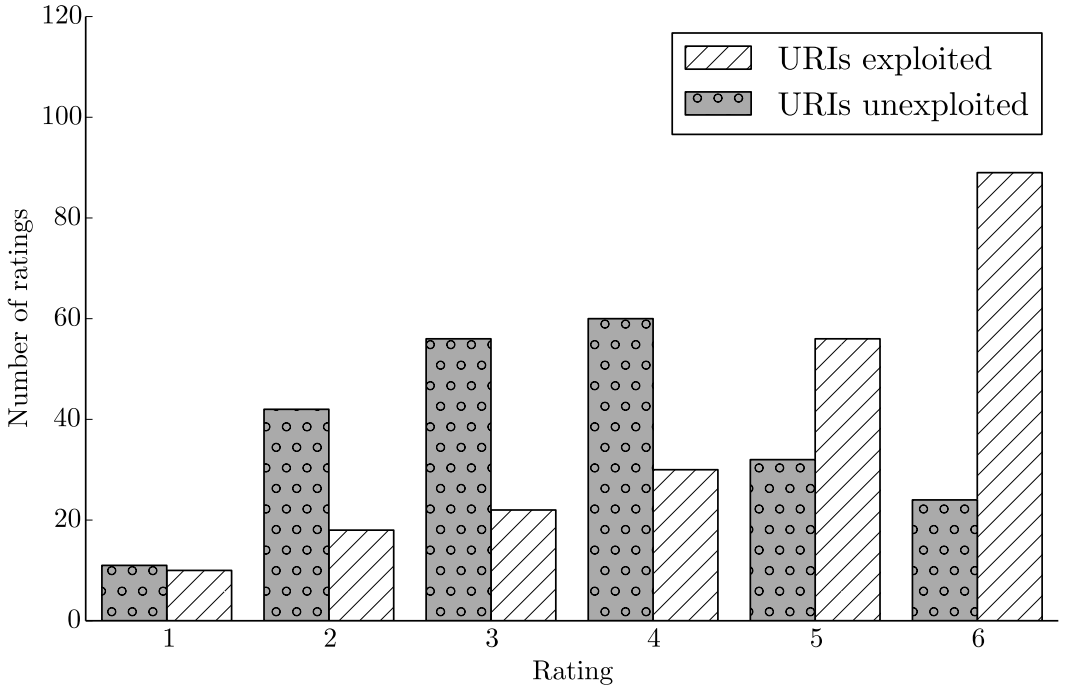


Figure 4.12: Responses to **Q5b/Q6b** "How would you rate the sentences, in terms of the **fluency** of the sentence?" The figure shows the absolute values of all responses by all participants to all sentence pairs.

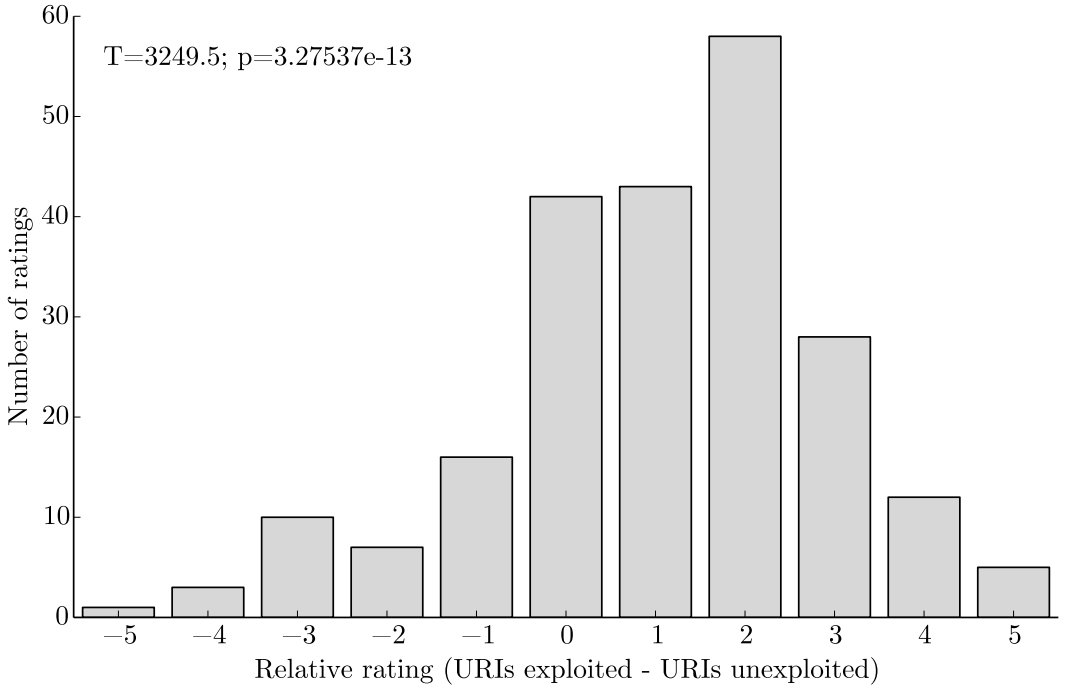


Figure 4.13: Responses to **Q5b/Q6b** "How would you rate the sentences, in terms of the **fluency** of the sentence?" The figure shows the relative values of all responses by all participants to all sentence pairs, (URIs exploited – URIs unexploited).

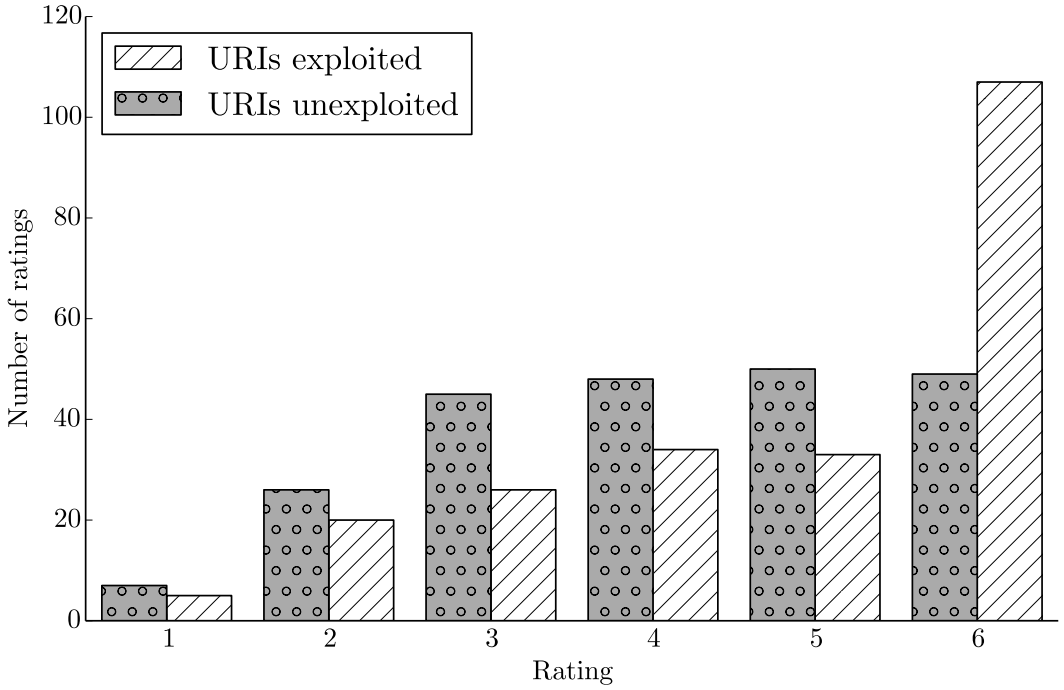


Figure 4.14: Responses to **Q5c/Q6c** "How would you rate the sentences, in terms of **how easily you can understand it**?" The figure shows the absolute values of all responses by all participants to all sentence pairs.

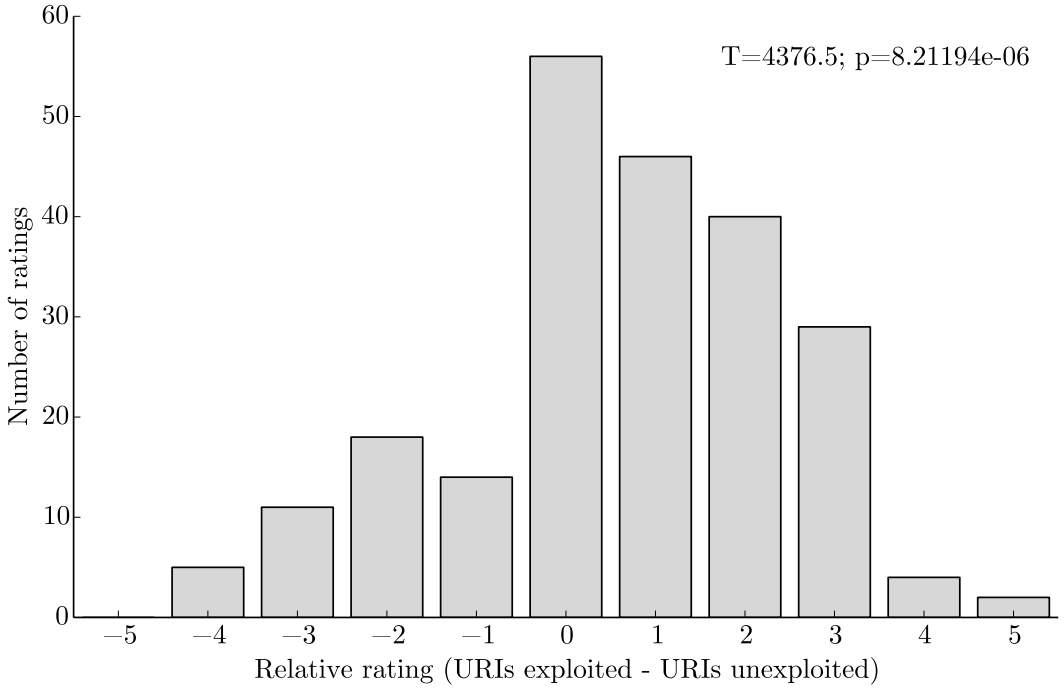


Figure 4.15: Responses to **Q5c/Q6c** "How would you rate the sentences, in terms of **how easily you can understand it**?" The figure shows the relative values of all responses by all participants to all sentence pairs, (URIs exploited – URIs unexploited).

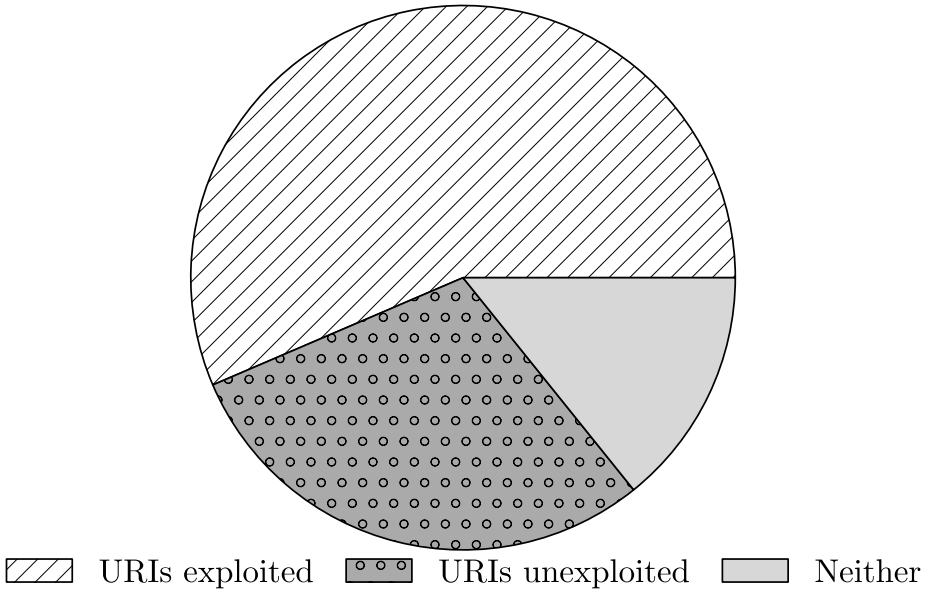


Figure 4.16: Responses to **Q7** "Which sentence do you think is the better explanation?" The figure shows the responses by all participants to all sentence pairs.

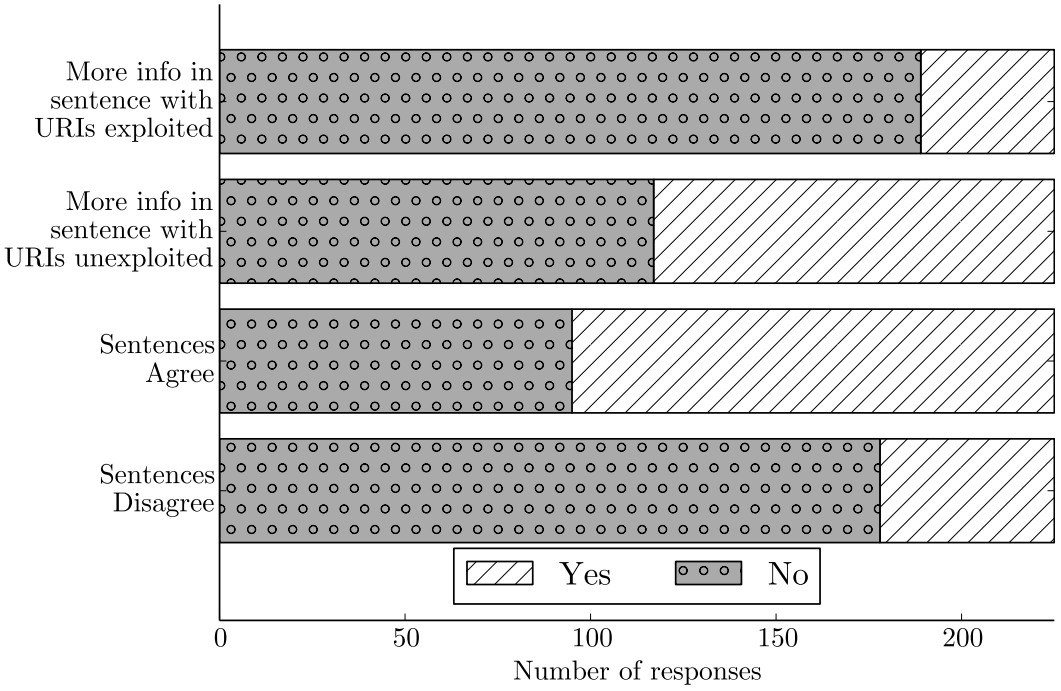


Figure 4.17: Responses to additional Yes/No questions **Q1/Q2, Q2/Q1, Q4, Q3**. The figure shows the responses by all participants to all sentence pairs.

Sentence Pair	Grammar	Fluency	Comprehension
1	Better	Better	Better
2	No difference	Better	Better
3	No difference	Better	Better
4	Better	Better	Better
5	Better	Better	Better
6	Worse	No difference	No difference
7	Better	Better	Better
8	Worse	Worse	Worse
9	Better	Better	Better
10	No difference	Better	Better
11	No difference	No difference	No difference
12	Better	Better	Better
13	No difference	No difference	No difference
14	No difference	Better	No difference
15	Worse	No difference	No difference
Total (B/ND/W)	+3 (6/6/3)	+9 (10/4/1)	+8 (9/5/1)

Table 4.6: The results of the study broken down by pair of sentences, showing when the sentences exploiting the linguistic information in URIs performed significantly better or worse, where $p < 0.05$.

Participant	Grammar	Fluency	Comprehension
1	Better	Better	Better
2	No difference	No difference	Worse
3	No difference	No difference	No difference
4	Better	Better	Better
5	No difference	No difference	No difference
6	No difference	Better	Better
7	Better	Better	Better
8	Better	Better	Better
9	No difference	Better	Better
10	No difference	Better	No difference
11	No difference	No difference	No difference
12	No difference	No difference	No difference
13	No difference	No difference	No difference
14	No difference	No difference	No difference
15	No difference	No difference	No difference
Total (B/ND/W)	+4 (4/11/0)	+7 (7/8/0)	+5 (6/8/1)

Table 4.7: The results of the study broken down by participant, showing when the sentences exploiting the linguistic information in URIs performed significantly better or worse, where $p < 0.05$.

4.5.2 Active voice

An extension of the fact that verbs can be extracted, in addition to nouns, is that where a subject and verb can both be extracted, it is possible to use the active voice, rather than the passive voice. The active voice is potentially more engaging to read, as well as being less verbose, and, outside of academia, is often seen as preferable.

4.5.3 Incorrect conjugation

In some cases, the realisation engine was not able to correctly conjugate the verbs, resulting in incorrect English. This is particularly true of some irregular verbs. For example, in Sentence 5 `patient was safed` instead of saved.

4.5.4 Unexpected realisation engine behaviour

Sentence 5 also illustrates another flaw that occurred in a number of the sentences. The realisation engine, when presented with a noun phrase head like `Patient 115 3`, appears to try to interpret the numbers as the quantity rather than just an identifier. The can also be seen in Sentence 15, where `agent2` becomes `2 agent`. It is possible that this behaviour can be suppressed by passing noun phrase heads as "canned text", but this would result in the loss of certain realisation features such as correct capitalisation.

4.6 Summary

This chapter has described a novel approach to utilising the linguistic information in URIs to enrich natural language explanations of provenance. By developing a technique to tokenise URIs, and assessing the performance of an off-the-shelf part-of-speech tagger to tag tokenised URIs, it has been shown that it is possible to create templates for generating natural language that exploit this information. Provenance explanations generated using these techniques have been shown to be more grammatically correct, more fluent, and easier for users to understand.

These techniques offer the prospect of being able to generate truly natural sounding explanations of provenance graph in a domain-independent way that was previously not possible, significantly extending the state-of-the-art. The following chapter builds upon this progress by attempting to understand how to put these sentences together into larger multi-sentential explanations.

Chapter 5

Structuring Provenance Explanations

Amongst other things, Chapters 3 and 4 described a system capable of generating natural language sentences from provenance graphs, and evaluated their performance as explanations. This chapter builds upon that work by building larger, multi-sentential explanations from medium-sized provenance graphs. One particular challenge of building larger texts from provenance graphs is that it is necessary to map a graphical structure to a linear text. That is to say, that there are a number of different approaches that could be used to convert a graphical structure into a linear one, each with its advantages and disadvantages. The focus of this chapter is to investigate a number of aspects of these approaches, and evaluate how they perform.

The chapter begins with an investigation where experts in using PROV were asked to explain ten provenance graphs, with a goal of understanding how experts would structure these explanations. It then describes an approach to structuring these explanations automatically, that was created using the results of this investigation.

To close, a user evaluation is conducted, exploring the importance of the direction of event ordering in these automatically generated explanations.

The primary contribution of this chapter is the insight it provides into how humans structure their own explanations from provenance graphs. Even though — as will be shown later in the chapter — it was clear humans structured their explanations according to the visual layout of the diagram they were presented with, there were still interesting

variations in the precise way they did this.

5.1 Humans explaining PROV

In order to understand how best to generate explanations from provenance graphs, a good initial step was to ascertain how provenance experts would achieve a similar task, under the assumption that human users would prefer explanations generated according to similar patterns to those used by the provenance experts. Consequently, an investigation was run with the primary objective of observing how provenance experts would structure their explanations, when asked to describe 10 different provenance graphs.

5.1.1 Methodology

The provenance graphs were obtained from the University of Southampton Provenance Store, and this investigation was carried out in accordance with University of Southampton ethics approval ERGO/FPSE/19658.

The participants were drawn from the University of Southampton Provenance Research Group, and consequently can all be considered experts in PROV and reading W3C Provenance Working Group diagrams (Provenance Working Group, 2011). The participants each took part in the exercise independently. As the data was drawn from the Southampton Provenance Store, and the participants from the Southampton Provenance Research Group, a number of the participants were responsible for creating some of the graphs used in the study. However, as the point of this study was to understand how an expert would explain a PROV graph, it was felt that this would only enhance the quality of the explanations.

At each stage of the exercise, a participant was shown a relatively small provenance graph (between 5 and 24 resources each) in the form of a W3C Provenance Working Group diagram. They had one minute to understand and familiarise themselves with the graph, and plan how they would describe it. After the minute had elapsed, they were given an unlimited amount of time to describe the graph, and their responses recorded as audio. The purpose of requiring a whole minute for the participant to familiarise themselves with the graph was to ensure that they had had sufficient time to examine the graph, instead of commencing their description prematurely. Consequently, it was not a concern that some of the participants had seen some of the graphs before.

The first graph shown to the participants was used to ensure that they understood the protocol, and the results from that graph are not included here. The participants were subsequently shown a sequence of ten graphs, whose results are shown here. The graphs were drawn from the University of Southampton Provenance Store, and were generated by a number of different users. They were selected to provide a variety of

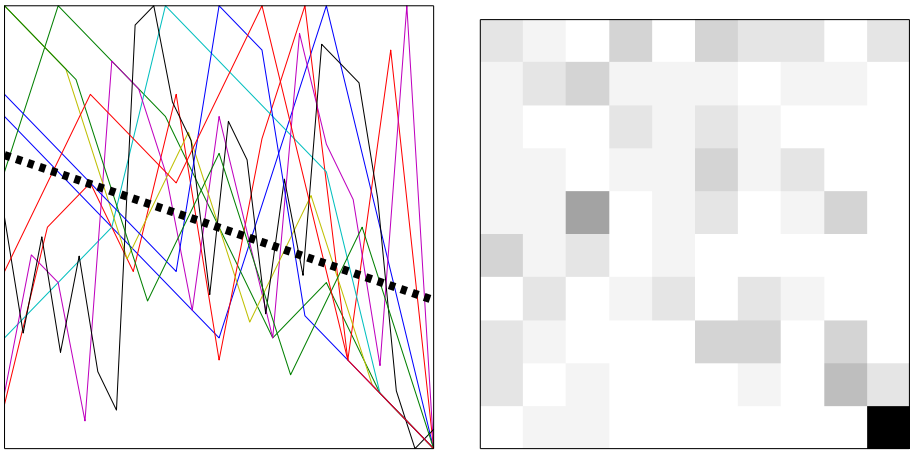
different shapes and sizes, to best explore how the experts would generate explanations from those different types of graph. As previously stated, the graphs contained between 5 and 24 resources each. There are larger provenance graphs on the Provenance Store, but this upper limit was set to keep the graphs at a legible size when printed on A3 paper. Additionally, it was felt that applications would most likely not want to generate explanations from graphs larger than this, without either choosing a more relevant subgraph or attempting to generate a summary explanation, rather than an all-encompassing one. Consequently, this study should be characterised as exploring the structuring of explanations from small- to medium-sized PROV graphs.

The responses of each participant were transcribed, to preserve anonymity, and are included in Appendix D, including the participants' responses to the initial example graph.

The participants could have been using one of two techniques for structuring their responses, or perhaps a combination of the two, or neither. The first approach, henceforth called the *visual-based approach*, takes a view that the graph should be read left-to-right from top-to-bottom in the same manner as an English language text. This would result, to a first approximation, in a chronologically ordered structure, because Provenance Working Group diagrams have the most recent resources at the bottom-right, and the oldest at the top-left. The second approach, henceforth called the *topology-based approach*, involves performing a topological sort on the provenance graph (or a reverse topological sort), resulting in concepts that are related in the graph appearing close together in the explanations. The algorithm for creating a document structure from a topology-based ordering is described in Section 5.1.2.

In order to determine which, if either, approach the participants were using, the graphs were coded up according to these two strategies. The difference between these two approaches can be seen in Figure 5.1. As is clear from this graph, there is only a weak anticorrelation between the two approaches ($r^2 = 0.11$).

Each transcript was coded up accordingly, where each code was inserted into the transcript at the first point a particular PROV resource was mentioned or strongly implied. If two resources were mentioned at exactly the same time, or in groups, the codes were inserted in the order most consistent with the rest of the explanation. For example, in the PROV graph shown in Figure 1.2, which was used as the training graph in this experiment (Graph 0), `ex:john` would be coded 1C, because it is first (1) in the visual layout, and third (C) in the topological sort. Likewise, `ex:cake` would be coded 8A, because it is eighth (8) in the visual layout, and first (A) in the topological sort. Consequently, if the participant's description was "John baked a cake.", this would be coded up as "John (1C) baked (7B) a cake (8A)." These codings are included in the transcripts in Appendix D.



(a) (b)

Figure 5.1: Mapping visual-based ordering to topology-based ordering. Horizontal axis: Visual-based ordering; Vertical axis: Topology-based ordering. (a) shows the trace of each graph in the study, comparing the topology-based ordering to the visual-based ordering. (b) shows a heatmap of these points, helping give a clearer indication of general trends, and in this case highlighting that the visual-based approach tends to end with the same resource that the topology-based approach starts with.

The coded responses to the exercise were then plotted visually to help identify any obvious patterns. In order to do this, the participants' responses first needed to be normalised, because different participants omitted different resources from their explanations, and the graphs were of varying size. This normalisation was done by taking a set of responses (e.g. `[1, 3, 5, 4]` — note `2` is missing), and first changing the values of each code so that all numbers between 1 and N were present, where N is the number of resources in the participant's response (e.g. `[1, 2, 4, 3]`). The responses were then scaled to be between 0 and 1 (e.g. `[0, 0.33, 1, 0.67]`). These responses form the vertical axis for each of the plots shown in Figures 5.3, 5.4, 5.5, 5.6, and 5.7. The horizontal axis is evenly spaced from 0 to 1 with $N-1$ intervals (e.g. `[0, 0.33, 0.67, 1]`), and represents the passage of time throughout the participant's response. Consequently, we are able to compare how the ordering of the participants' responses relate to each type of ordering.

Similarly, in order to detect correlation, a linear regression was performed, and is plotted on each graph as a thick black dashed line. This process also yields an r^2 value, indicating the strength of the correlation, and a p value, indicating how likely it is that there is in fact no correlation (which in this case is the null hypothesis).

5.1.2 Topology-based ordering

Whilst the use of Rhetorical Structure Theory (Mann and Thompson, 1988) is common within the NLG community, its use in the context of generating explanations of PROV graphs is somewhat limited. This is because RST contains many different relationships that can describe, in theory, any form of text, but PROV graphs simply record events that happened, including who and what were involved. In RST this would simply require the *elaboration*, *list*, and *sequence* relationships, and consequently would only use a small, relatively inconsequential part of the theory. As a result of this, the use of RST was not considered for structuring explanations generated from PROV graphs. However, as is discussed in more detail in Chapter 6, RST may be of use in structuring natural language summaries of PROV graphs, and is consequently discussed more there.

Instead, a topology-based algorithm was implemented as a potential candidate for structuring longer multi-sentential texts, as shown in Figure 5.2. One downside of this approach is that a topological sort can only be performed on acyclic graphs, whereas PROV graphs can, in some cases, have cycles. Were this approach to be pursued further, some technique would need to be introduced for handling these cycles, however only a few of the valid PROV graphs in the dataset contained cycles, so this was not a problem at this stage.

5.1.3 Expectations

As this was an exploratory investigation, the expectations were somewhat vague, but could be expressed as follows:

Expectation 1 There would be some correlation between both the visually-based orderings and the topology-based orderings and the participants' responses.

Expectation 2 There would be no marked difference between the responses of the different participants.

5.1.4 Results

The results are shown in Table 5.1, and Figures 5.3, 5.4, 5.5, 5.6, and 5.7.

Result 1 Overall, there was a stronger correlation between the visual-based orderings and the participants' responses than the topology-based ordering ($r^2 = 0.09$ and $r^2 = 0.00$ respectively). Contrary to **Expectation 1**, there was no discernible correlation between the topology-based ordering and the participants' responses, and the correlation between the visual-based ordering and the participants' responses was very weak.

Result 2 Contrary to **Expectation 2**, three of the participants seemed to adopt a chronological approach to ordering the graph, resulting in correlations where $r^2 = 0.45$, $r^2 = 0.41$, and $r^2 = 0.44$. Conversely, the remaining two participants started with the resulting PROV resource and then explained how that resource was achieved using a combination of chronological and anti-chronological approaches, resulting in anticorrelations where $r^2 = 0.18$ and $r^2 = 0.25$ (See Table 5.1). These can be seen visualised in Figure 5.6.

Result 3 The difference between the two sets of participants described in **Result 2** can be seen particularly clearly in Figure 5.4i. The fact that there are possibly two different overlapping clusters in these results has possibly reduced the observed correlations. Consequently, the correlations were recomputed for the two groups separately. As can be seen in Table 5.1, the group containing participants 1, 2, and 4 showed a correlation between the visual-based ordering and the participant responses where $r^2 = 0.51$, and the group containing participants 3 and 5 showed an anticorrelation where $r^2 = 0.21$. As for the topology-based ordering, both groups showed weak relationships with an anticorrelation where $r^2 = 0.05$ and a correlation where $r^2 = 0.07$ respectively. This does, however, support **Expectation 1** that both of these orderings would correlate to the participant responses. Nevertheless, the visual-based ordering clearly has the stronger correlation.

	Visual-based ordering		Topology-based ordering		Stronger
	r^2	p	r^2	p	correlation
Graph 1	0.00	1.00	(0.03)	0.445	(Topology) $*$
Graph 2	0.16	0.006	(0.18)	0.004	(Topology)
Graph 3	0.03	0.256	0.01	0.462	Visual $*$
Graph 4	0.05	0.203	0.05	0.204	Visual $*$
Graph 5	0.07	0.045	0.13	0.003	Topology
Graph 6	0.00	0.969	(0.02)	0.398	(Topology) $*$
Graph 7	0.74	2.74×10^{-20}	0.00	0.643	Visual
Graph 8	0.21	0.001	(0.06)	0.110	Visual
Graph 9	0.02	0.479	(0.11)	0.537	(Topology) $*$
Graph 10	0.04	0.341	(0.01)	0.686	Visual $*$
Participant 1	0.45	2.36×10^{-15}	(0.04)	0.049	Visual
Participant 2	0.41	2.84×10^{-12}	(0.01)	0.355	Visual
Participant 3	(0.18)	3.28×10^{-5}	0.05	0.027	(Visual)
Participant 4	0.44	1.11×10^{-14}	(0.08)	0.004	Visual
Participant 5	(0.25)	1.65×10^{-5}	0.18	0.0003	(Visual)
P1, P2, and P4	0.51	1.61×10^{-45}	(0.05)	1.21×10^{-4}	Visual
P3 and P5	(0.21)	6.91×10^{-9}	0.07	0.0001	(Visual)
Overall	0.09	1.08×10^{-10}	0.00	0.265	Visual

Table 5.1: The results of the human explanation investigation, showing the strengths of correlations (r^2), and the probability that there was really no correlation (p). Brackets denote an anticorrelation. In the "Stronger correlation" field, an asterisk ($*$) signifies that neither ordering showed a statistically significant correlation, where $p < 0.05$.

Inputs: Provenance graph G, Unordered set of sentences S
Outputs: Ordered list of sentences P

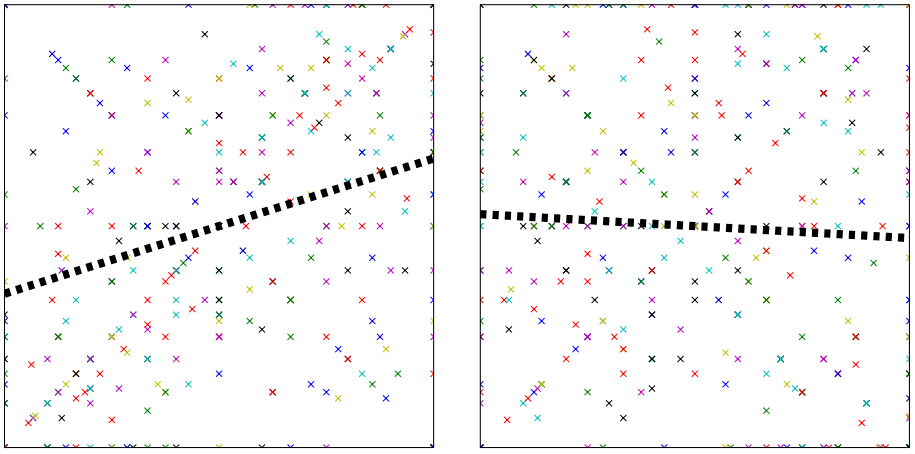
1. Start with a provenance graph G, and a set of sentences S, generated according to the method described in Chapter 3. Each sentence s_i in S has a coverage set C_i, containing the triples covered by that sentence. C_i' is the flattened version of the coverage set, containing only the subjects and objects from C_i, as a single 1-dimensional vector. The empty ordered list P represents the paragraph.

2. A topological sort is performed on G resulting in an ordered list of nodes N.

3. For node n_j in N:

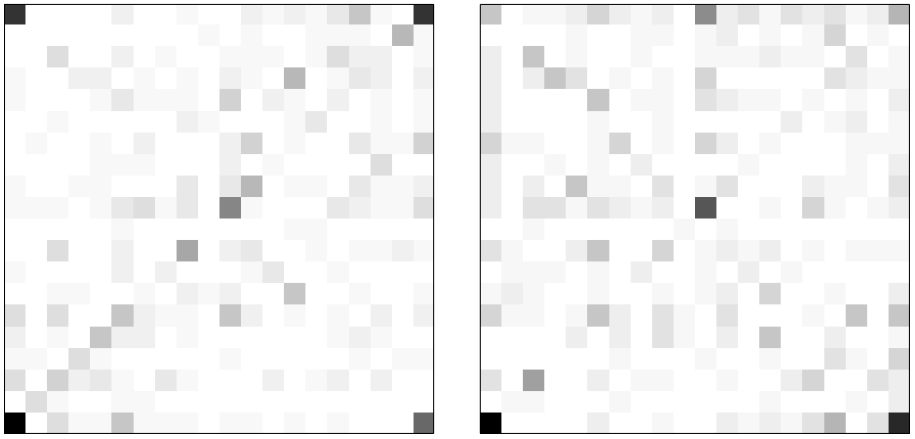
   - For sentence s_k in S, if n_j is in C_k:
     (a) Add s_k to P.
     (b) Remove s_k from S

Figure 5.2: An algorithm for structuring a multi-sentential text based on a topological sort.



(a) Visual-based ordering

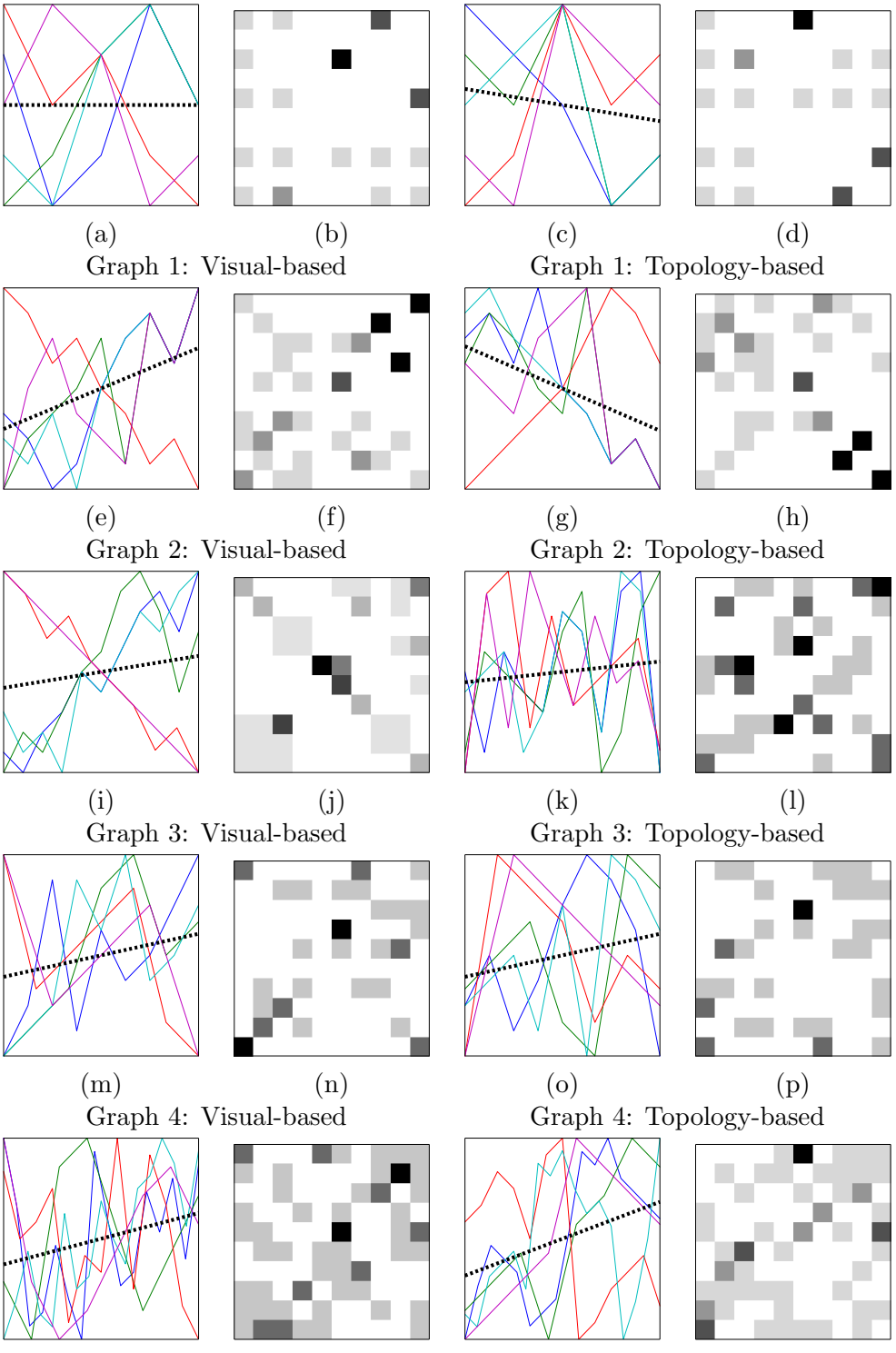
(b) Topology-based ordering



(c)

(d)

Figure 5.3: Participant responses to all graphs, from all participants, shown as scatter graphs, and heatmaps to more clearly show how many responses exist at each location. Linear regression shown as thick black dashed line.



(a) (b) (c) (d)
Graph 1: Visual-based Graph 1: Topology-based

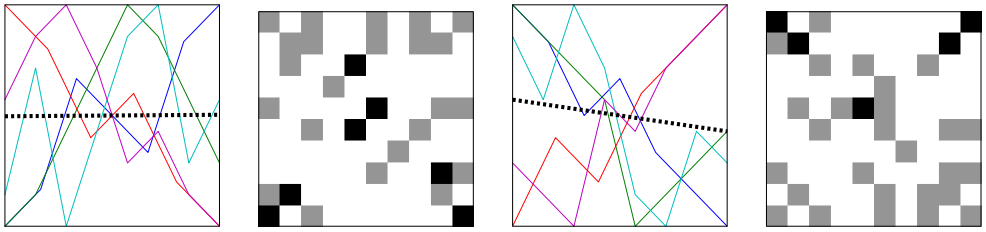
(e) (f) (g) (h)
Graph 2: Visual-based Graph 2: Topology-based

(i) (j) (k) (l)
Graph 3: Visual-based Graph 3: Topology-based

(m) (n) (o) (p)
Graph 4: Visual-based Graph 4: Topology-based

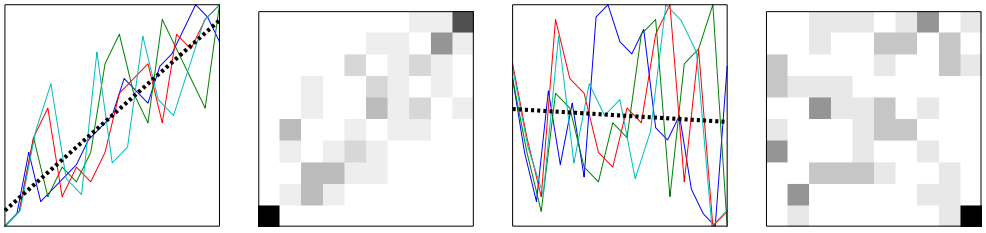
(q) (r) (s) (t)
Graph 5: Visual-based Graph 5: Topology-based

Figure 5.4: Participant responses broken down by graph (Graphs 1–5). Linear regression shown as thick black dashed line. Of particular note, is (i), which very clearly illustrates the difference between the participants structuring their responses chronologically, and those structuring them anti-chronologically



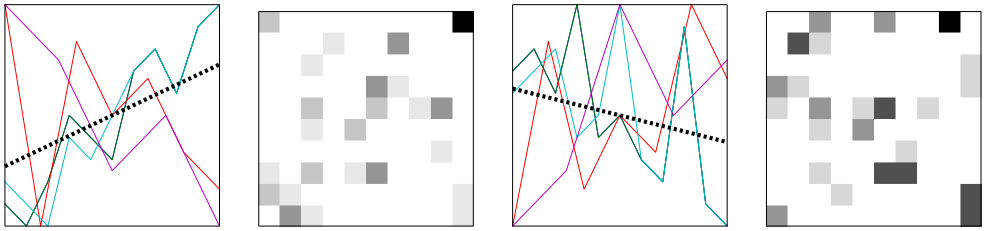
(a) (b) (c) (d)

Graph 6: Visual-based Graph 6: Topology-based



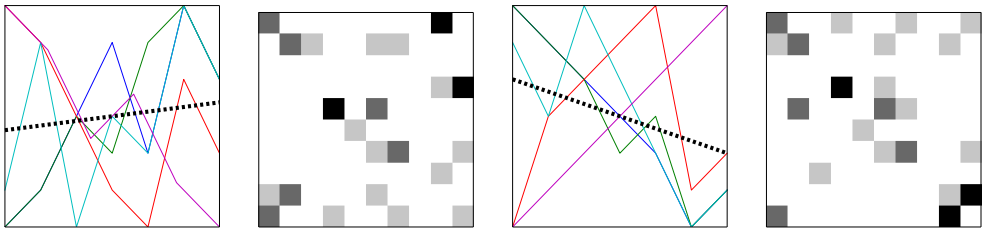
(e) (f) (g) (h)

Graph 7: Visual-based Graph 7: Topology-based



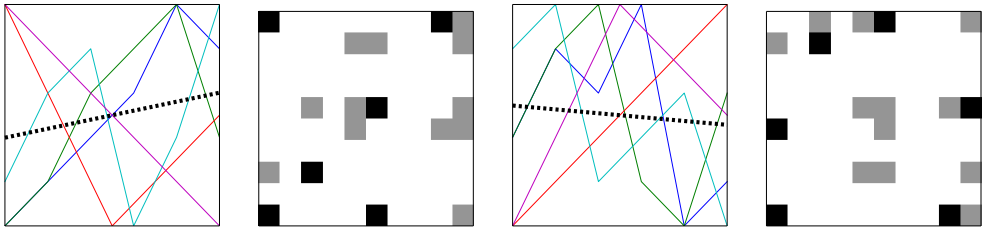
(i) (j) (k) (l)

Graph 8: Visual-based Graph 8: Topology-based



(m) (n) (o) (p)

Graph 9: Visual-based Graph 9: Topology-based



(q) (r) (s) (t)

Graph 10: Visual-based Graph 10: Topology-based

Figure 5.5: Participant responses broken down by graph (Graphs 6–10). Linear regression shown as thick black dashed line. Of particular interest is (e), which shows the responses to Graph 7 as compared to the visual-based layout. This is the strongest correlation in the participant responses, perhaps evoked by the fact that it was also the largest graph.

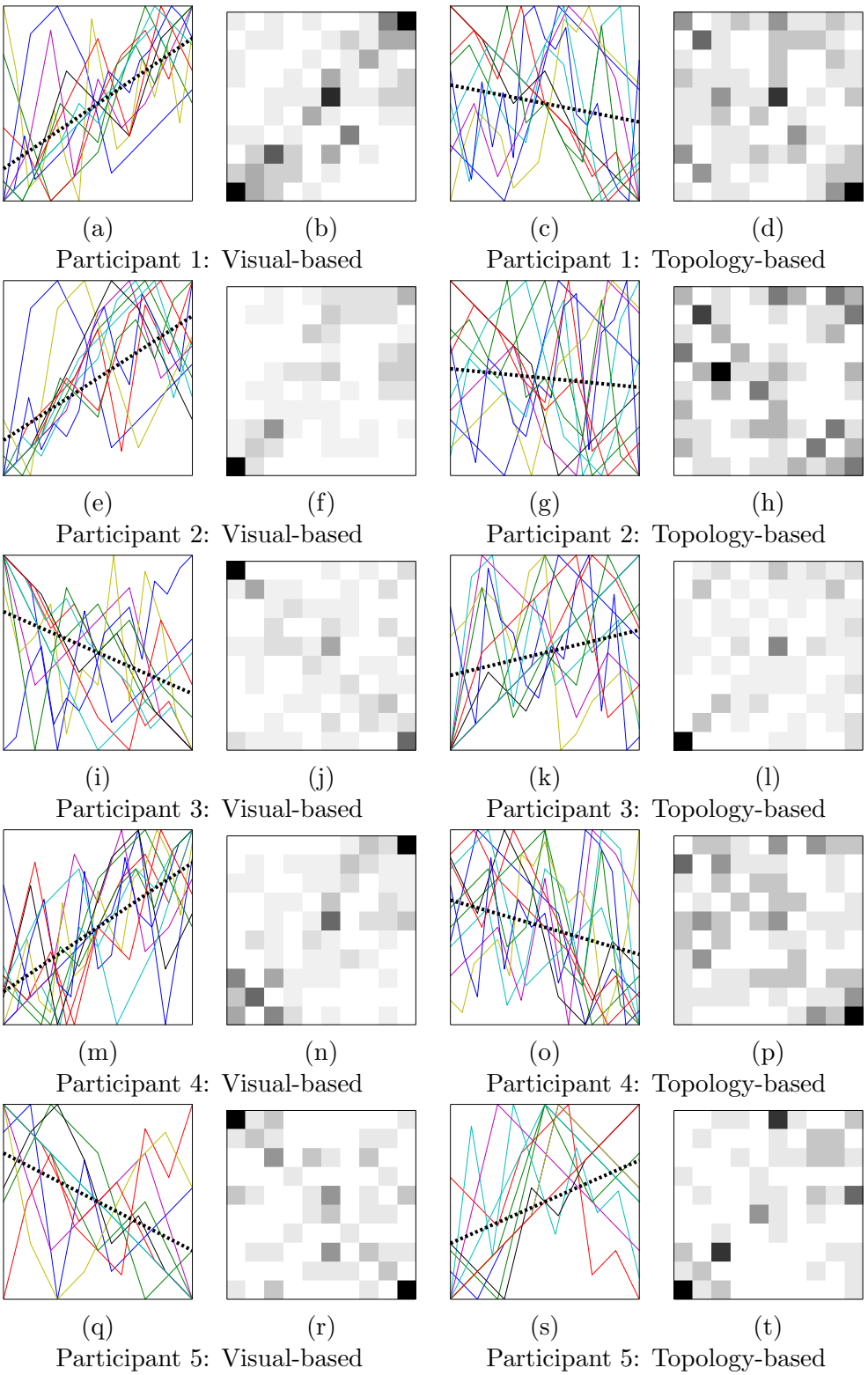
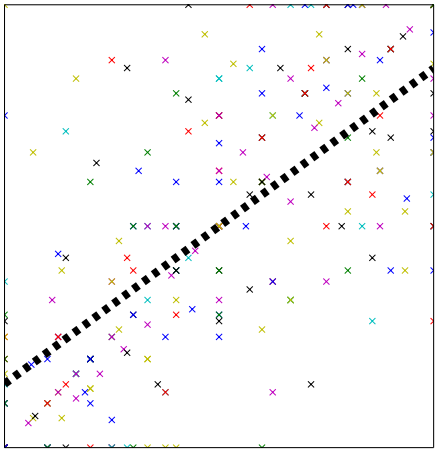
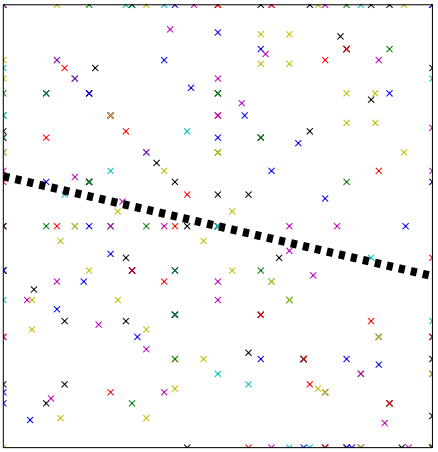


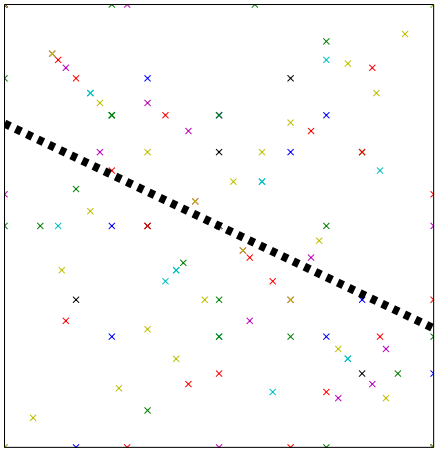
Figure 5.6: Participant responses broken down by participant. Linear regression shown as thick black dashed line. Of particular interest is the fact that all participants demonstrate a relatively strong correlation to the visual-based ordering, but that the direction of this correlation varies between participants.



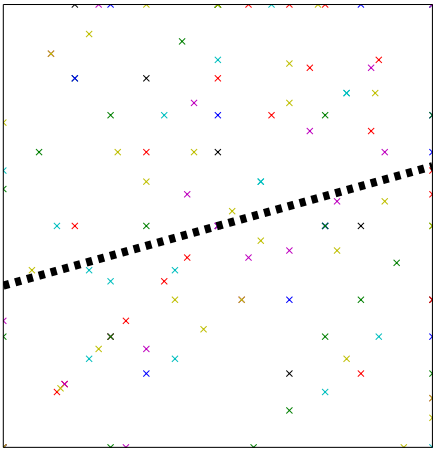
(a) P1, P2, P4: Visual-based
$r^2 = 0.51$



(b) P1, P2, P4: Topology-based
$r^2 = (0.05)$



(a) P3, P5: Visual-based
$r^2 = (0.21)$



(b) P3, P5: Topology-based
$r^2 = 0.07$

Figure 5.7: Participant responses broken down by participant into two groups (P1, P2, P4 and P3, P5). Linear regression shown as thick black dashed line.

5.2 An example generation

Section 5.3 will describe an experiment to help understand users' preferences with respect to the structure of automatically generated explanations. The purpose of this section is to provide an example of how the paragraphs used in that experiment were generated. This generation is based on a similar graph to that used in the examples in Chapters 3 and 4 to aid comprehension. An explanation generated from the provenance graph of which this is a subgraph was later used in the experiment in Section 5.3, and can be seen in Appendix C.

The small PROV graph used in this example generation is shown in Figure 5.9, and consists of six provenance resources: 1 agent, 2 entities, and 3 activities. The resulting generation is shown in Figure 5.8, and consists of four sentences. The flow of information throughout the pipeline can be seen in Figure 5.10, indicating the various stages of processing each piece of information goes through to produce a natural language provenance explanation.

> Derek compiled chart 1.
> Chart 1 was compiled at 2012-03-02T10:30:00+00:00.
> Derek illustrated chart 1 from composition.
> Derek composed composition.

Figure 5.8: The paragraph generated from the example graph, ordered according to the topological sort.

This generation used an implementation of the architecture introduced in Chapter 3. Consequently, it began by applying a number of templates — including the template used as an example in Chapter 3 — over the graph. This resulted in a set of (template, bindings) pairs. These were then whittled down using the sentence selection algorithm described in Section 3.3, leaving four (template, bindings) pairs, including the pair from the example in Chapter 3. The ordering algorithm based on a topological sort, shown in Figure 5.2, was then used to sort the sentences. As `ex:chart1` is the only node in this graph with no incoming relations, it was the first node in the topological sort. Consequently, the sentences with `ex:chart1` appear first in the ordering. The final sentence in this generated paragraph only involved the agent `ex:derek`, the activity `ex:compose`, and the entity `ex:composition`, and consequently appeared lower in the topological sort, leading this sentence to be the last in the paragraph. This ordered list of sentences was then passed to the realisation engine, which produced the paragraph surface text. These processes can be seen summarised in Figure 5.10.

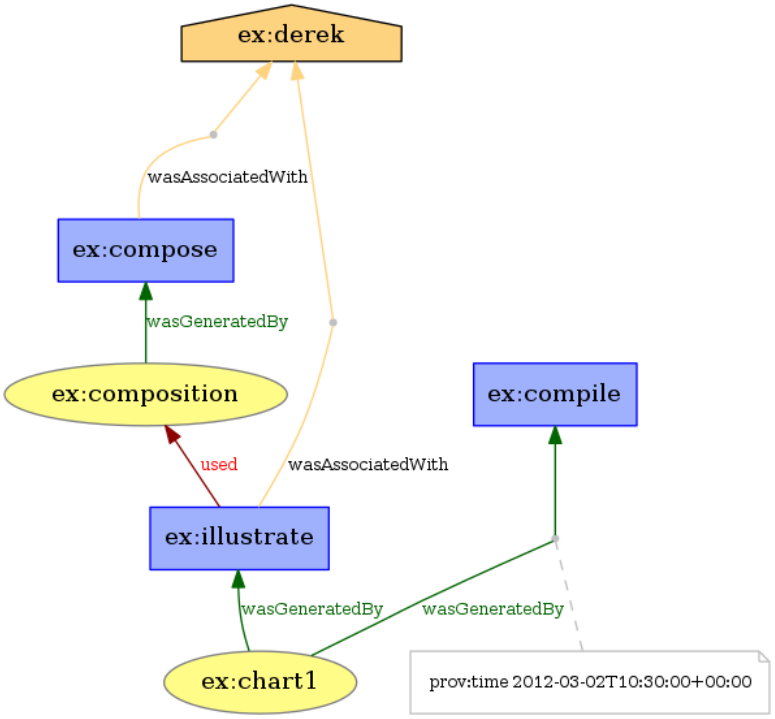


Figure 5.9: The example graph used for the generation in Section 5.2.

5.3 Exploring the effect of chronology

In the experiment at the beginning of this chapter, the experts who participated were split fairly evenly (3 to 2) with respect to whether they structured their responses chronologically or antichronologically. The purpose of the experiment in this section is to determine which, if either, of these options is preferred by casual readers.

5.3.1 Methodology

The methodology for this user evaluation was broadly the same as the evaluation in Section 4.4, and was carried out under the terms of University of Southampton ethics approval ERGO/FPSE/20104. The differences were that instead of sentence pairs, the participants were shown pairs of paragraphs. In each pair, each paragraph contained the same information, but one was sorted according to the topological sort algorithm described in Section 5.1.2, and one was sorted in the reverse order. To be clear, the two paragraphs in each pair contained exactly the same sentences, only in a different order. These two different orderings were used to simulate chronological and antichronological orderings. (Topological Sort $\sim$ Anti-chronological ordering; Reverse topological sort $\sim$ Chronological ordering.) Additionally, instead of being shown 15 pairs, the participants were each shown 10 pairs of paragraphs to keep the amount of time required from each participant reasonable.

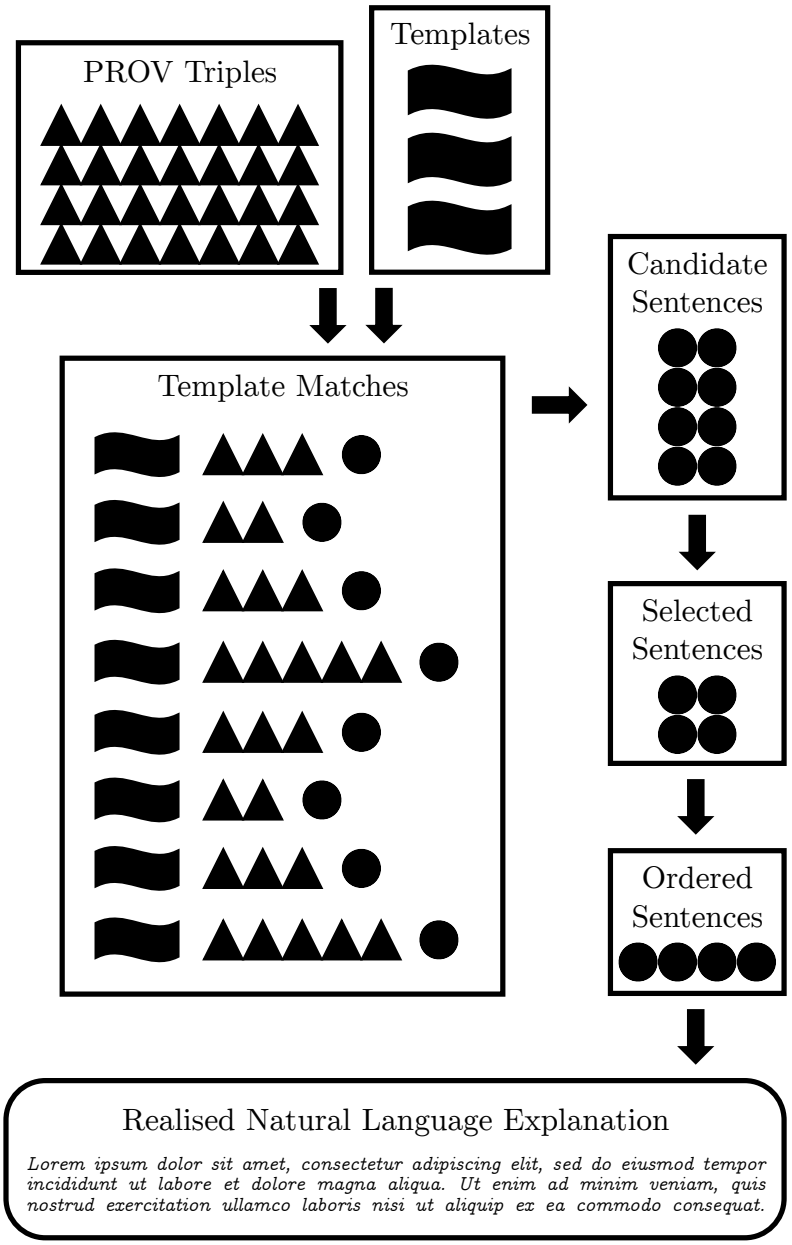


Figure 5.10: This figure shows the flow of information through the various stages of the pipeline. The templates are applied to the PROV document to generate sets of bindings — template matches — each of which represents a possible sentence. Each of these matches has an associated set of triples as its coverage set, which is used to whittle down the candidate sentences using the algorithm from Section 3.3. The ordering algorithms from this chapter are then used to select a sentence ordering, which is realised as a natural language document.

The graphs from which the paragraphs were generated were taken from six different applications on the Provenance Store, with each application representing between one and four sets of paragraphs. Nine of the graphs appear to have been generated in an automated fashion, whereas one appears to have been handcrafted. As shown in Table 5.2, these graphs were selected to represent a range of different sized graphs, generating explanations with from 6 to 26 sentences each. Whilst the Provenance Store naturally contains graphs that would have generated explanations with a greater number of sentences, these were excluded for the same reason as in the experiment conducted earlier in Section 5.1. As previously stated, it was felt that any application wanting to generate an explanation from provenance would most likely either want to explain a small subgraph of a large provenance graph, or alternatively generate a summary explanation. As discussed in Chapter 3, the former task of choosing a subgraph from which to generate an explanation falls under the NLG process of content determination, and is outside of the scope of this work. Similarly, the latter task of generating summary explanations is also left for further work, and discussed in Chapter 6. Consequently, in this case also, this experiment explores the structuring of explanations generated from small- to medium-sized provenance graphs only. The paragraphs generated for this experiment are shown in Appendix C.

Paragraph pair #	**Num. of sentences**
1	6
2	6
3	16
4	9
5	26
6	9
7	16
8	15
9	11
10	12

Table 5.2: Number of sentences per paragraph, ranging from 6 to 26. The mean number of sentences per paragraph was 12.6, and the median was 11.5.

The participants used the same locally hosted Web-based interface as in Chapter 4, but some of the questions were different, to account for the different nature of the task. The participants were asked the following questions:

Q1 Do the paragraphs disagree at all? (Yes / No)

Q2 How would you rate Paragraph A, in terms of:

Q2a the logical ordering of events? (1 *Very Bad* – 6 *Very Good*)

Q2b the fluidity of the description? (1 *Very Bad* – 6 *Very Good*)

Q2c how easily you can understand it? (1 *Very Bad* – 6 *Very Good*)

Q3 How would you rate Paragraph B, in terms of:

Q3a the logical ordering of events? (1 *Very Bad* – 6 *Very Good*)

Q3b the fluidity of the description? (1 *Very Bad* – 6 *Very Good*)

Q3c how easily you can understand it? (1 *Very Bad* – 6 *Very Good*)

Q4 Which paragraph do you think is the better explanation? (Paragraph A / Paragraph B / Neither)

In total, 18 participants took part in this evaluation. Similarly to the experiment in Chapter 4, these were drawn from the Electronics and Computer Science academic unit of the University of Southampton, with the same language proficiency criteria applied.

5.3.2 Expectations

Unlike the evaluation in Chapter 4, it was not desirable for either condition — chronological ordering or anti-chronological ordering — to be found to be preferable to the other. However, the following statements were being tested:

Expectation 1 Participants, in aggregate, would have a clear preference for one ordering over the other.

Expectation 2 There should be no reported disagreement between the paragraphs in each pair — after all, they are just the same paragraph with the sentences reversed.

5.3.3 Results

Result 1 Contrary to **Expectation 1**, only one of the three dimensions showed a statistically significant difference between the topological sort and reverse topological sort (where $p < 0.01$): logical ordering. There was no significant difference in terms of the fluidity or the comprehensibility of the paragraphs under the different conditions. These results can be seen in Figures 5.11, 5.12, 5.13, 5.14, 5.15, and 5.16.

Result 2 The median response for the paragraphs ordered according to a topological sort was 3, 3, and 3.5 for the dimensions of logical ordering, fluidity, and comprehensibility respectively. The median response for the paragraphs ordered according to a reverse topological sort was 4 for each of the three dimensions. The median difference in response for each pair of paragraphs was 0 across all dimensions. These results do not support **Expectation 1**.

Result 3 For the paragraphs ordered according the topological sort, the mode response was 3, 4, and 4 for logical ordering, fluidity, and comprehensibility respectively. For the paragraphs ordered according to the reverse sort, the mode response was 5, 4, and 5 for logical ordering, fluidity, and comprehensibility respectively. The mode difference between responses was 0 for every dimension. Again, these results do not support **Expectation 1**.

Result 4 10.6% of participants reported that the paragraphs disagreed. It is unclear why this number is so high, though it is perhaps notable that one participant answered 'yes' for this question (**Q4**) for every paragraph pair, which accounts for 53% of those responses. However, given that 89.4% of the time participants said that the paragraphs did not disagree, this lends support to **Expectation 2**, if not agreeing with it entirely. This can be seen in Figure 5.18.

Result 5 When asked directly, the participants stated 26.1% of the time that they thought the paragraph generated using the topological sort was the better explanation. Conversely, 42.8% of the time the participant reported that they thought the paragraph generated using the reverse topological sort was the better explanation. Neither paragraph was considered better 31.1% of the time. This can be seen in Figure 5.17, and does not seem to support **Expectation 1**.

5.4 Summary

This chapter has explored a number of techniques to linearising provenance graphs so as to be able to generate multi-sentential explanations from them. First, exploring how human experts describe provenance graphs, it was shown that humans presented with a diagram of a provenance graph tend to describe it in a similar order to how it is laid out on the page. However, it was also shown that some participants prefer to start from the top left, and work down to the bottom right, while other participants prefer to start with the conclusion of the graph, and then explain how that situation was arrived at. This investigation also showed a correlation to a topology-based ordering, though this correlation was considerably smaller than that to the visual-based ordering.

Because the approaches taken by the two groups of participants — starting at the top-left, or starting at the bottom-right — map relatively closely with the chronology of the graph, and because there was a fairly even split between the number of participants who took either approach, it was desirable to determine whether casual readers would exhibit a preference for texts generated accordingly. In Section 5.3, a user evaluation was carried out to attempt to establish whether using this algorithm to sort events chronologically or anti-chronologically performed better in terms of logical ordering, fluidity, and comprehensibility. The results showed that the paragraphs ordered chronologically (i.e. the

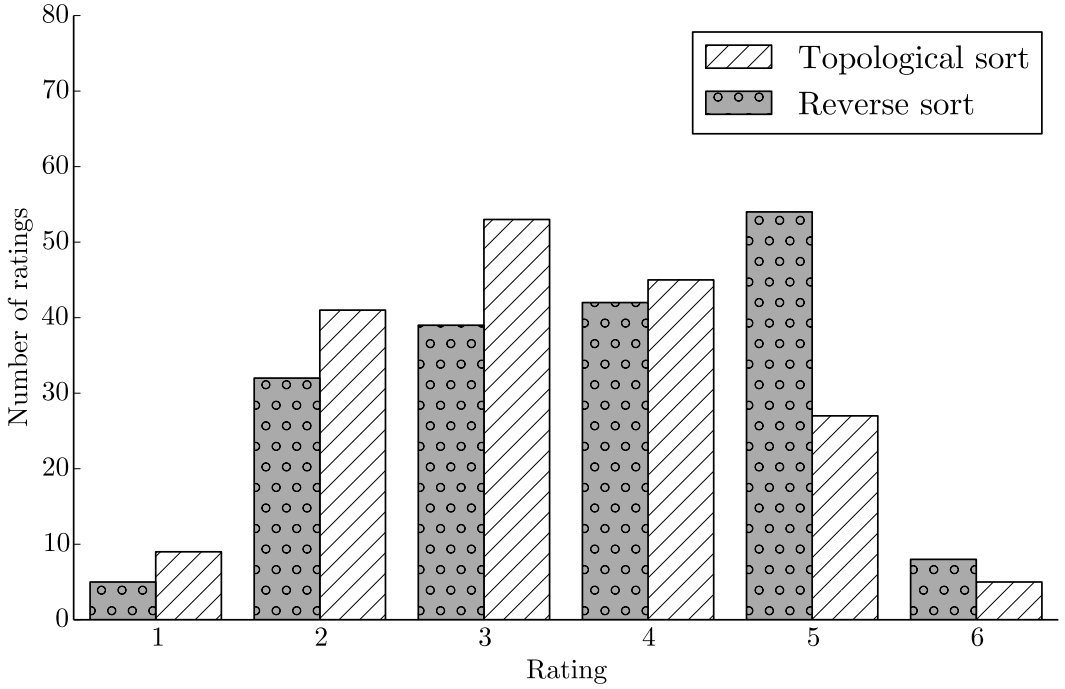


Figure 5.11: Responses to **Q2a/Q3a** "How would you rate the paragraphs, in terms of the **logical ordering** of events?" The figure shows the absolute values of all responses by all participants to all paragraph pairs.

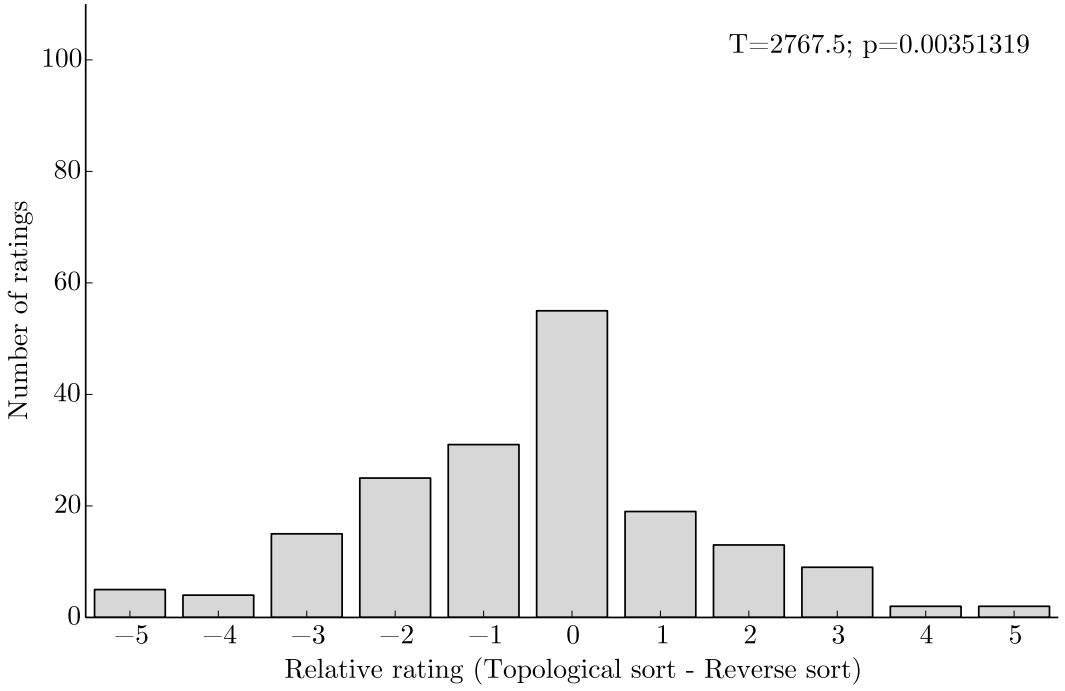


Figure 5.12: Responses to **Q2a/Q3a** "How would you rate the paragraphs, in terms of the **logical ordering** of events?" The figure shows the relative values of all responses by all participants to all paragraph pairs, (Topological sort – Reverse sort).

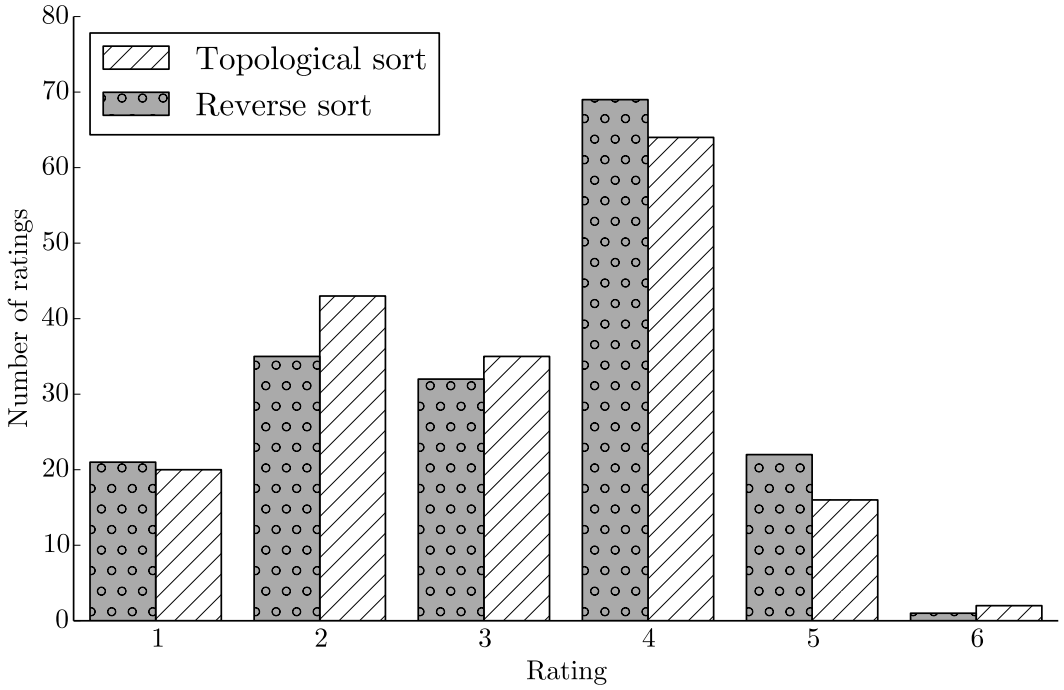


Figure 5.13: Responses to **Q2b/Q3b** "How would you rate the paragraphs, in terms of the **fluidity** of the description?" The figure shows the absolute values of all responses by all participants to all paragraph pairs.

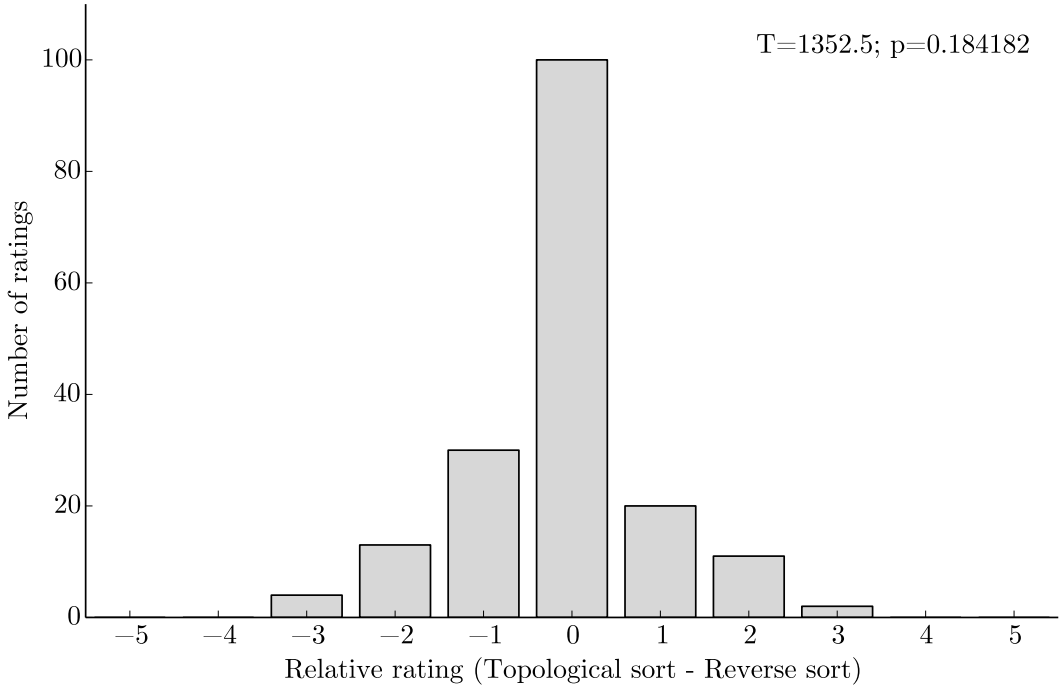


Figure 5.14: Responses to **Q2b/Q3b** "How would you rate the paragraphs, in terms of the **fluidity** of the description?" The figure shows the relative values of all responses by all participants to all paragraph pairs, (Topological sort – Reverse sort).

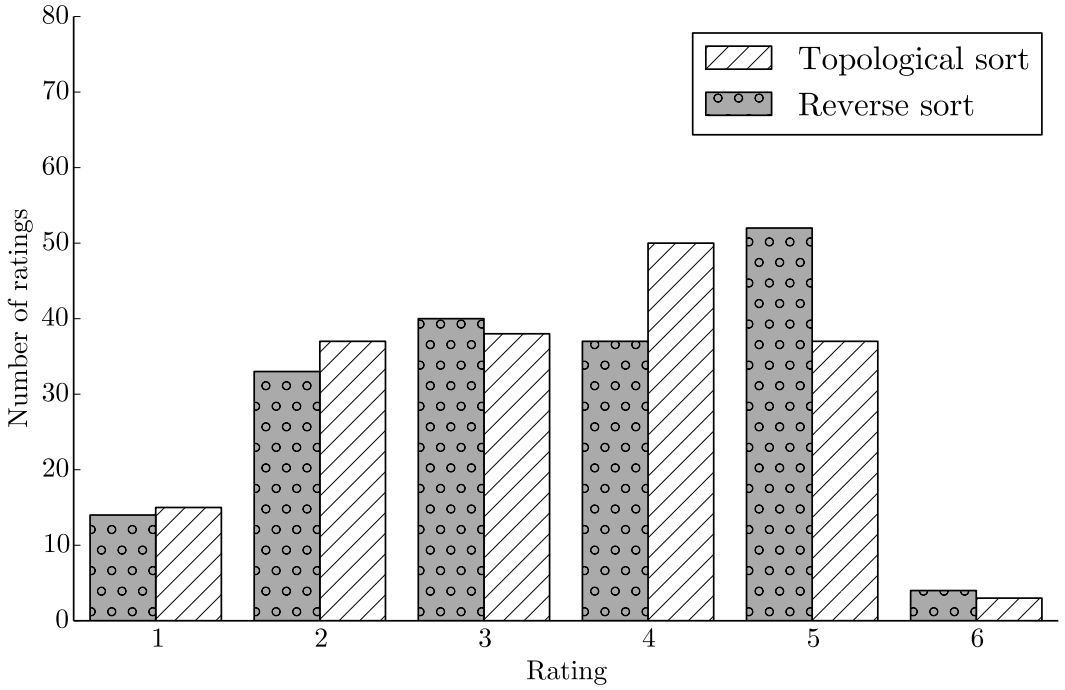


Figure 5.15: Responses to **Q2c/Q3c** "How would you rate the paragraphs, in terms of **how easily you can understand it**?" The figure shows the absolute values of all responses by all participants to all paragraph pairs.

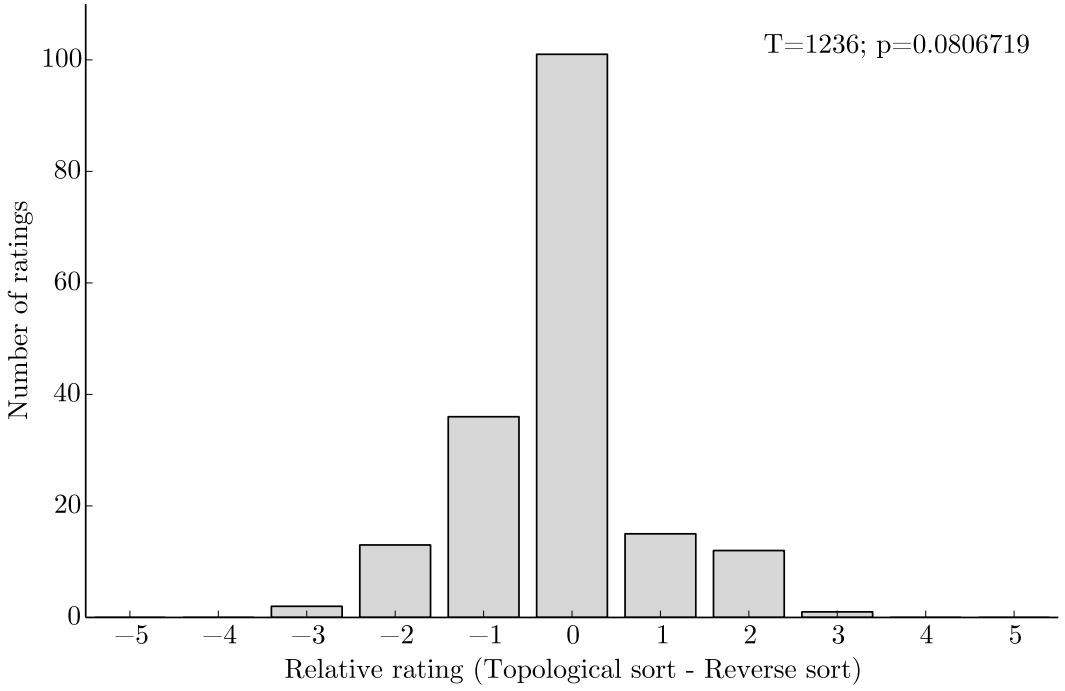


Figure 5.16: Responses to **Q2b/Q3b** "How would you rate the paragraphs, in terms of **how easily you can understand it**?" The figure shows the relative values of all responses by all participants to all paragraph pairs, (Topological sort – Reverse sort).

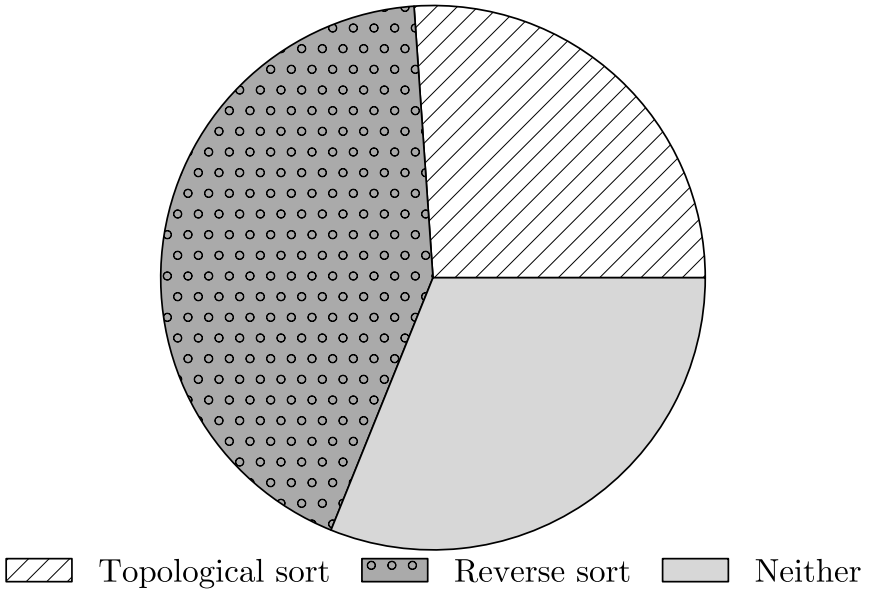


Figure 5.17: Responses to **Q4** "Which paragraph do you think is the better explanation?" The figure shows the responses by all participants to all paragraph pairs.

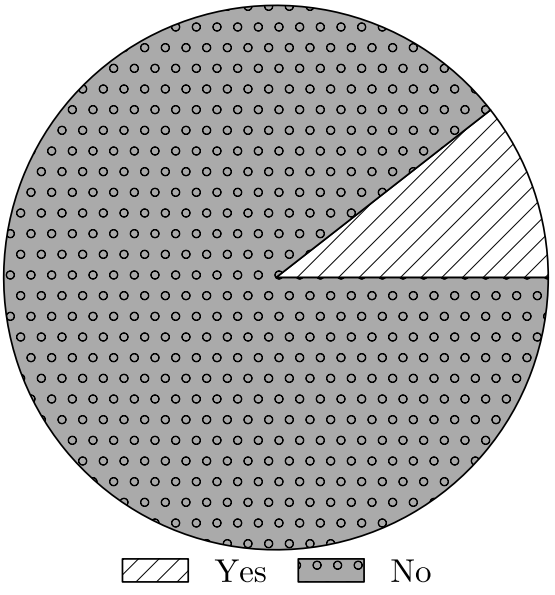


Figure 5.18: Responses to **Q1** "Do the paragraphs disagree at all?". The figure shows the responses by all participants to all paragraph pairs.

reverse topological sort) were considered marginally more logical than those that were not, however the remainder of the results were statistically insignificant.

This would suggest that in future, systems should be built that, by default, show PROV graphs in a chronological order, though more work needs to be done in this area to establish exactly how this ordering should be accomplished.

Chapter 6

Conclusions and Future Work

This dissertation has described significant contributions to the state-of-the-art with regard to the domain-independent generation of natural language explanations from provenance graphs. The purpose of this chapter is to review the most significant of these contributions, summarising how they build upon the state-of-the-art, and reiterating why those contributions are of value (see Section 6.1). In addition, this chapter also posits a number of areas that would be ripe for exploration, should this work be continued further (see Section 6.1.5). The chapter closes with a brief summary describing the overall potential impact of this work, in Section 6.3.

6.1 Original Contributions

In its effort to generate natural language explanations from provenance graphs in a fashion independent of the subject domain, this work has created a number of original contributions to the state-of-the-art. It has expanded upon existing NLG theories, applying these technologies in new ways; it has demonstrated the viability of exploiting the linguistic information encoded in the URIs denoting PROV resources; it has demonstrated that this approach significantly improves the quality of the natural language explanations generated from provenance graphs; it has investigated how humans structure their own natural language explanations from provenance graphs; and it has explored how chronology affects the perceived quality of these natural language explanations. These contributions are explored in more depth in the remainder of this section.

6.1.1 Novel application of NLG technology

Chapter 3 proposed an adaptation of the existing 'consensus' NLG architecture that would allow for template-based generation of natural language explanations from provenance graphs. Using templates or rules to generate sentences is nothing particularly new

within the NLG community, but this novel adaptation recognises that templates perform roles that cross the traditional module boundaries of document planning, microplanning, and realisation, whilst in many cases maintaining interoperability with existing software components. For example, because these templates produce text specifications, rather than surface texts, advanced off-the-shelf realisation technologies can be used, further leveraging the pre-existing research of the NLG community.

Furthermore, this work also introduced the comparison of the challenge of sentence selection to the set-cover problem. The task of identifying all the possible sentences that could be generated from a provenance graph by a given set of templates is simply a matter of executing as many queries as there are templates. The realised text specification of each template paired with a set of bindings expresses a part of the graph in natural language — and can be said to cover that subgraph. That subgraph, in turn, can be considered to be a set of triples, which is a subset of the triples that make up the entire provenance graph. This subset has been called the *coverage set*, and if sentences are chosen from the set of all possible sentences so that the union of their coverage sets is equal to the set of triples that makes up the whole provenance graph, it can be ensured that the whole graph has been expressed in the generated text. Chapter 3 demonstrated an algorithm, derived from a greedy solution to the set-cover problem, that was able to select a set of sentences guaranteed to cover an entire provenance graph, should it be possible for the templates to do so.

The significant contribution of the work in this chapter was to adapt an architecture, and build a system, that facilitates the generation of natural language explanations from provenance graphs, whilst being able to utilise off-the-shelf NLG technologies.

6.1.2 Exploitation of linguistic information in URIs

Exploiting the linguistic information in PROV URIs to enrich natural language explanations had never previously been attempted. Similar efforts have previously tended to focus on ontologies, rather than instance data, and consequently Chapter 4 considerably extends the state-of-the-art by demonstrating that even using off-the-shelf NLP tools, it is possible to extract linguistic information from URIs in a way that could lead to richer natural language explanations.

Chapter 4 tackled the challenges of tokenisation and tagging. Tokenisation is the process whereby a URI denoting a PROV resource is split into its various linguistic component, and the pattern-based approach presented here was able to correctly tokenise over 95% of the URIs in one particular dataset. Tagging, subsequently, is the process of identifying what role a particular linguistic token plays in the text — that is, it is a verb, or an adjective, or something else — and the off-the-shelf tagger that was used was able to correctly tag 92.3% of URI tokens. Whilst this performance was shown to be slightly

worse than a relatively simple tagger, it did crucially perform much better at identifying verbs — which are very important for enriching natural language explanations — correctly classifying 55% of the tokens in the dataset that were really verbs as verbs, whereas the relatively simple tagger mentioned previously would not be able to correctly identify any verbs at all.

The contribution of this section of Chapter 4 is that it allows for the creation of templates that would be able to generate richer natural language sentences from provenance graphs, without introducing the need for domain-specific linguistic information to be built in to the templates, or to be formally encoded in the RDF. One limitation of this particular contribution is that it relies of the linguistic information being both present in the URIs and extractable. In cases where this is not the case, the sentences produced are of the same kind as in the existing state-of-the-art.

6.1.3 Improvement of the quality of generated explanations

Having written templates for the system created in Chapter 3 that were able to make use of the linguistic information exploited earlier in Chapter 4, the next step was to evaluate the performance of these sentences with human users. This evaluation showed that the sentences generated exploiting this linguistic information from URIs, when compared to sentences unable to exploit this information — that is, the pre-existing state-of-the-art — outperformed those sentences significantly in terms of grammatical correctness, fluency, and comprehensibility. When asked directly which sentence they preferred, the participants responded that they preferred the sentence exploiting the linguistic information 56.4% of the time, compared to only 14.2% for the other sentences.

The contribution of this section of Chapter 4 was to demonstrate that the improvements made in Chapter 3 and earlier in Chapter 4 were not merely academic, but had a measurable impact on the perceived quality of the natural language explanations that it is possible to generate from provenance graphs, at least amongst the participants of the experiments. With a sample size of 15, all drawn from the Electronics and Computer Science academic unit, it is unclear how well these results would generalise to the population as a whole.

6.1.4 Exploration of how humans structure provenance explanations

Many provenance graphs contain more information than can reasonably be expressed in a single sentence. The architecture from Chapter 3 coupled with the techniques from Chapter 4 is able to generate these multiple sentences, but before multi-sentential natural language explanations can be generated from provenance graphs, a method first needs to be developed that would allow for these explanations to be structured sensibly.

As with many artificial intelligence problems, the solution was possibly to be discovered in the way humans accomplish the same task. Accordingly, an investigation was constructed and carried out looking at how humans structure their explanations of provenance graphs. This small-scale investigation found that when presented with a small- to medium-sized provenance graph in diagrammatic form, provenance experts would create a narrative that correlated most strongly with the visual layout of the graph — with an r^2 value as high as 0.51 for one group of participants. One other finding of this investigation was that participants were split fairly evenly between those who favoured a chronological ordering or an anti-chronological ordering of events. However, it should be noted that this experiment was conducted with only five participants, all from the same research group. It is consequently unclear if these results would generalise to all PROV experts.

The principal contribution of this section of Chapter 5 was in helping to understand how experts explain provenance graphs, and in discovering this split between the way the two groups of participants choose to order events.

6.1.5 Investigation of chronology on the quality of explanations

With the participants of the previous investigation split 3–2 on the issue of whether to order provenance graphs chronologically or anti-chronologically, a user study was performed to understand how these two approaches would fare across three dimensions: logical ordering; fluidity of description; and comprehensibility. Similarly to the results of the previous investigation earlier in Chapter 5, the participants were fairly evenly split on which of the two approaches they preferred, with neither approach gaining a majority of support, and with neither paragraph considered the better explanation 31.1% of the time.

The only dimension where there was a statistically significant difference was in respect to the logical ordering of events, where the more chronological approach to structuring the paragraphs was considered more logical. With respect to the other two dimensions, however, there was no statistically significant difference, and the mode difference between the responses to the two different paragraphs was zero for all three dimensions. Similarly to the experiment from Chapter 4, this study had a relatively small number of participants (18 in this case), all of whom were drawn from the Electronics and Computer Science academic unit. This means that it is unclear how well these results would generalise to the population as a whole.

The contribution of this section of Chapter 5 was in helping to understand that chronological ordering is preferred more often than the anti-chronological, which would prove useful in developing similar systems in the future.

6.2 Limitations and Future Work

Whilst this work is of clear importance, and has made a number of significant contributions, it is still very much at an early stage in development. With appropriate levels of resourcing, and sufficient time, there are a number of areas where this technology could be explored further:

6.2.1 Studies with more representative participants

One of the main criticisms of this work may well be the fact that the experiments were all conducted with a relatively small, and possibly unrepresentative sample of the population. The user studies from Chapters 4 and 5 had 15 and 18 participants respectively. Additionally, all of the participants were drawn from the University of Southampton's Electronics and Computer Science academic unit, and mostly either possessed, or were pursuing doctoral degrees. Consequently, even though many of them had no prior knowledge of PROV or even the Semantic Web, they were very likely more technologically aware than many of the target users of such a system would be.

Similarly, the PROV expert study from Chapter 5 had only 5 participants, all of whom were drawn from the Southampton University Provenance Working Group. In addition to the small sample size, it is possible that the fact that all the participants were drawn from the same research group has skewed the results. In order to assess whether these results hold true amongst all PROV experts, a much wider experiment would need to be conducted.

Were this work to be taken further towards exploitation, one might first want to verify that the results presented in this work hold for the population of interest. That could entail either performing research evaluating the system's effectiveness for a particular set of users within a particular application domain, or ideally by assessing its effectiveness amongst the general population. The former was not an option for this particular work because the goal was to find a domain-independent solution. The latter was excluded because of resource constraints, as such an experiment would have been prohibitively expensive and time consuming to run.

6.2.2 Improved grammatical correctness and fluency

As shown by the results of the experiments from Chapters 4 and 5 — not to mention a subjective reading of the generated sentences in Appendices B and C — one of the clear limitations of this work is that the sentences generated are not always of the highest quality English. That is to say, whilst the participants from the experiment in Chapter 4 clearly felt that the approach used was significantly more grammatical than

existing template-based approaches, in many cases the quality of the sentences could be improved.

One potential solution might be to use an off-the-shelf grammar checker to detect sentences that are grammatically incorrect. However, as the sentences are only realised at the final stage of the pipeline, it would be necessary to address how one would replace the faulty sentences. A solution to this problem might be to insert a stage into the algorithm described in Figure 3.1, whereby the sentence is realised after Step 7, and if the generated sentence is flagged as ungrammatical, reject it, repeating Step 7 until a grammatical sentence is found. Nevertheless, it is unclear how well such a grammar checker would work in this case, as a single sentence would provide limited context for the grammar checker.

When incorrect sentences are generated, it is often because the process of extracting the linguistic information from the URI has resulted in faulty information, or information that the realisation engine cannot process correctly — for example, if a noun is extracted from a URI but tagged as a verb, then the realisation engine might incorrectly be asked to treat that noun as a verb, which will only be possible in certain limited circumstances. This is only a problem with the templates that attempt to exploit the linguistic information in URIs. Consequently, another way to potentially use a grammar checker to improve the quality of the generated texts would be to generate the whole text, pass it into the grammar checker — helping provide the checker with more context — and replacing any ungrammatical sentences. The ungrammatical sentences would be replaced with text generated by the corresponding template that does not attempt to exploit linguistic information in URIs — which, whilst producing lower-quality English in general, do appear to result in better-quality English in the cases where the more complex templates produce clearly ungrammatical sentences.

6.2.3 Improved structure and PROV Summaries

This work only addresses the generation of natural language explanations from standard provenance graphs, and does so in such a way that every part of the graph is converted regardless of how large or repetitive the graph might be. In such cases, this would similarly result in natural language explanations that were large and repetitive, and not at all engaging to a potential reader.

Nevertheless, existing work by Moreau (2014) demonstrates an approach to reducing provenance graphs by identifying repeated constructs within those graphs, and reducing it down into a more manageable size using only a few extensions to the PROV vocabulary. These summaries potentially open the door to the development of more sophisticated explanation systems that are able to not only simply convert a provenance graph into text, but that are able to make quantitative assessments about the graph, such as, for

example, "Most ride plans were accepted, however a minority were rejected. Of the rejections 86% were rejected by a rider, and 14% by a driver." Sentences such as this are clearly well beyond what is achievable with the existing system, but could be possible if the system was extended to make use of these PROV summaries.

In addition to potentially creating more engaging texts, such an approach would be more likely to be able to take advantage of the features of RST to help structure the texts (Mann and Thompson, 1988). This is because these summaries would be more likely to include more of the relationships described by RST, such as contradictions or comparisons, as opposed to the relatively few that would have been applicable in this work. Whilst the templates that were created for the experiments presented in this dissertation cover all the concepts in PROV, some of these concepts could be handled significantly better through the use of, for example, RST. For instance, specialisations and revisions are difficult to accomplish with a purely templated approach, due to the need to place the sentences relative to one another in time through the use of the pluperfect tense, or similar linguistic techniques. RST could help provide the information needed to do this, whilst retaining the templated approach.

6.2.4 Improving extraction of linguistic information

In addition, there are alternative sources from which the linguistic information needed to perform the sophisticated realisation performed in this work could be extracted. Whilst the URI is the only linguistic feature guaranteed to be present in a PROV graph — which is why it was focussed on in this work — there are other features like `rdfs:label` and `foaf:name` which are present in many cases, and could potentially be exploited in a similar fashion to the way URIs were used in this work. During the investigation at the beginning of Chapter 5, on many occasions the participants made use of features like these when describing the graphs, as well as making use of `rdf:type`/`prov:type` to talk about resources as being a member of a class, in a way that has not been attempted here. These can both be seen in the transcripts in Appendix D.

Finally, there needs to be more investigation into how it would be possible to automatically assess how good a generated sentence is. At present, there are a number of errors that can occur, either in the process of extracting the information from URIs, or in the realisation stage of generation. These often lead to grammatically incorrect, potentially misleading sentences; given that it is always possible to fall back to the simpler templates that do not attempt to exploit informally encoded linguistic information, having a better idea of when that would be appropriate should help to increase the overall quality of the explanations generated.

6.2.5 Referring expressions and aggregation

Taking another look at the consensus NLG architecture, as well as the modified version presented in Chapter 3, there are a number of areas of that architecture that are yet to be fully explored with respect to how they apply to generating natural language explanations from provenance graphs. In particular, the submodules of "Referring expression generation" and "Aggregation" have yet to be tackled in any meaningful way.

Referring expression generation would allow for the use of anaphora (i.e. it, that, those, which, etc.), and shortened versions of names to be used to refer to concepts. This would allow for the shortening of explanations, and also increase the amount of variation in the text, helping to make it more engaging. Meanwhile, aggregation would allow for multiple sentences to be aggregated into one, larger sentence, leading to texts that feel more coherent and less disjointed.

An improvement in either of these two areas would lead to more engaging, shorter texts being generated, that would probably be more suitable for presenting to casual users of a provenance-enabled system.

6.3 Closing Summary

In a future economy powered by Linked Data, systems will not only need to provide provenance for their own decisions and data processing, but will have to explain their use of data from external Web-enabled systems not necessarily under their control. To this end, PROV has been standardised as the interoperable model of provenance for use on the Web. Whilst figures and diagrams will continue to provide a valuable interface to such provenance data, there will exist a need for systems to be able to articulate this information using natural language, as this is the medium with which their users will be the most familiar.

This chapter has reviewed the progress made in this regard in the proceeding chapters, as well as presenting a number of avenues for future research, should this work be continued.

The combination of the techniques developed within this work, building individual sentences and then building them into a larger explanations, allows for the domain-independent expression of provenance graphs as natural language explanations, capable of communicating rationale and, with extension, even presenting cogent arguments to users, heralding the era of the social machine.

Appendix A

University of Pennsylvania POS tags

These are the tags output by the part-of-speech tagger used in Chapter 4. For each tag, the expanded linguistic description is provided, followed by a number of examples of tokens that would match that tag. The descriptions and examples are copied verbatim from NLTK's built-in documentation module (Bird et al., 2009), which can be accessed inside python with the following instruction, provided the appropriate NLTK bundles have been installed: `nltk.help.upenn_tagset()`. Tags that did not occur in the ProvStore URI dataset are excluded from this list.

-NONE- zero
: 0 none no zero

CC conjunction, coordinating
: & 'n and both but either et for less minus neither nor or plus so therefore times v. versus vs. whether yet

CD numeral, cardinal
: mid-1890 nine-thirty forty-two one-tenth ten million 0.5 one forty-seven 1987 twenty '79 zero two 78-degrees eighty-four IX '60s .025 fifteen 271,124 dozen quintillion DM2,000

DT determiner
: all an another any both del each either every half la many much nary neither no some such that the them these this those

FW foreign word
: gemeinschaft hund ich jeux habeas Haementeria Herr K'ang-si vous lutihaw alai je jour objets salutaris fille quibusdam pas trop Monte terram fiche oui corporis

IN preposition or conjunction, subordinating
astride among uppon whether out inside pro despite on by throughout below within for towards near behind atop around if like until below next into if beside

JJ adjective or numeral, ordinal
third ill-mannered pre-war regrettable oiled calamitous first separable ectoplasmic battery-powered participatory fourth still-to-be-named multilingual multidisciplinary

JJR adjective, comparative
bleaker braver breezier briefer brighter brisker broader bumper busier calmer cheaper choosier cleaner clearer closer colder commoner costlier cozier creamier crunchier cuter

JJS adjective, superlative
calmest cheapest choicest classiest cleanest clearest closest commonest corniest costliest crassest creepiest crudest cutest darkest deadliest dearest deepest densest dinkiest

LS list item marker
A A. B B. C C. D E F First G H I J K One SP-44001 SP-44002 SP-44005 SP-44007 Second Third Three Two * a b c d first five four one six three two

MD modal auxiliary
can cannot could couldn't dare may might must need ought shall should shouldn't will would

NN noun, common, singular or mass
common-carrier cabbage knuckle-duster Casino afghan shed thermostat investment slide humour falloff slick wind hyena override subhumanity machinist

NNP noun, proper, singular
Motown Venneboerger Czestochwa Ranzer Conchita Trumplane Christos Oceanside Escobar Kreisler Sawyer Cougar Yvette Ervin ODI Darryl CTCA Shannon A.K.C. Meltex Liverpool

NNPS noun, proper, plural
Americans Americas Amharas Amityvilles Amusements Anarcho-Syndicalists Andalusians Andes Andruses Angels Animals Anthony Antilles Antiques Apache Apaches Apocrypha

NNS noun, common, plural
undergraduates scotches bric-a-brac products bodyguards facets coasts divestitures storehouses designs clubs fragrances averages subjectivists apprehensions muses factory-jobs

POS genitive marker
' 's

PRP pronoun, personal
hers herself him himself hisself it itself me myself one oneself ours ourselves ownself self she thee theirs them themselves they thou thy us

PRP$ pronoun, possessive
her his mine my our ours their thy your

RB adverb
occasionally unabatingly maddeningly adventurously professedly stirringly prominently technologically magisterially predominately swiftly fiscally pitilessly

RBR adverb, comparative
further gloomier grander graver greater grimmer harder harsher healthier heavier higher however larger later leaner lengthier less-perfectly lesser lonelier longer louder lower more

RP particle
aboard about across along apart around aside at away back before behind by crop down ever fast for forth from go high i.e. in into just later low more off on open out over per pie raising start teeth that through under unto up up-pp upon whole with you

SYM symbol
% & ' " ".)). * + ,. < = > @ A[fj] U.S U.S.S.R * ** ***

TO "to" as preposition or infinitive marker
to

VB verb, base form
ask assemble assess assign assume atone attention avoid bake balkanize bank begin behold believe bend benefit bevel beware bless boil bomb boost brace break bring broil brush build

VBD verb, past tense
dipped pleaded swiped regummed soaked tidied convened halted registered cushioned exacted snubbed strode aimed adopted belied figgered speculated wore appreciated contemplated

VBG verb, present participle or gerund
telegraphing stirring focusing angering judging stalling lactating hankerin' alleging veering capping approaching traveling besieging encrypting interrupting erasing wincing

VBN verb, past participle
multihulled dilapidated aerosolized chaired languished panelized used experimented flourished imitated reunifed factored condensed sheared unsettled primed dubbed desired

VBP verb, present tense, not 3rd person singular
predominate wrap resort sue twist spill cure lengthen brush terminate appear tend stray glisten obtain comprise detest tease attract emphasize mold postpone sever return wag

VBZ verb, present tense, 3rd person singular
bases reconstructs marks mixes displeases seals carps weaves snatches slumps stretches authorizes smolders pictures emerges stockpiles seduces fizzes uses bolsters slaps speaks pleads

Appendix B

Sentence pairs

These are the sentence pairs that were used in the human evaluation in Chapter 4. For each sentence pair, there are two sentences that were generated from the same PROV subgraph. In each case, one sentence was generated exploiting the linguistic information informally encoded in the URIs, whilst the other was generated without this additional information.

Sentence Pair 1

URIs exploited:
`Reputation manager generated opinion 1.`

URIs unexploited:
`‘/rs/reputation_manager’ generated ‘/rs/opinion/1/’ by ‘/reputationapi/#generate_opinion_1’.`

It should perhaps be noted that in this example, the verb ‘generated’ has actually been extracted from the URI ‘/reputationapi/#generate_opinion_1’.

Sentence Pair 2

URIs exploited:
`Application 1 was posted.`

URIs unexploited:
`‘/#application/1/’ was generated by ‘/reputationapi/#post_123’.`

Sentence Pair 3

URIs exploited:
`Opinion 2 was generated.`

URIs unexploited:
`‘/rs/opinion/2/’ was generated by ‘/reputationapi/#generate_opinion_1’.`

Sentence Pair 4

URIs exploited:
`Rm posted application 1.`

URIs unexploited:
`‘/rm’ generated ‘/#application/1/’ by ‘/reputationapi/#post_123’.`

Sentence Pair 5

URIs exploited:
`Transporter 92 transported 106 1 radioactive.`

URIs unexploited:
`‘/data/121212/transporter92’ did ‘/data/121212/TransportRadioactive106.1’.`

Sentence Pair 6

URIs exploited:
`115 3 patient was safed at 2012-12-12T14:24:09.129000+00:00.`

URIs unexploited:
`‘data/121212/SafePatient115.3’ was generated by`
`‘data/121212/SafeDropPatient115.1’ at 2012-12-12T14:24:09.129000+00:00.`

Sentence Pair 7

URIs exploited:
`Execution step 4808 used building 949 2 at 2012-02-07T13:36:32+00:00.`

URIs unexploited:
‘/data/ExecutionStep4808’ used ‘/data/Building949.2’ at 2012-02-07T13:36:32+00:00.

Sentence Pair 8

URIs exploited:
Vote 1043 0 was executioned step at 2011-12-18T01:00:17+00:00.

URIs unexploited:
‘/data/UpVote1043.0’ was generated by ‘/data/ExecutionStep652’ at 2011-12-18T01:00:17+00:00.

Sentence Pair 9

URIs exploited:
The ingredients were butter, eggs, flour and sugar.

URIs unexploited:
‘/#ingredients’ had the members ‘/#butter’, ‘/#eggs’, ‘/#flour’ and ‘/#sugar’.

Sentence Pair 10

URIs exploited:
John baked cake.

URIs unexploited:
‘/#john’ derived ‘/#cake’ by ‘/#baking’.

Sentence Pair 11

URIs exploited:
Composition was composed.

URIs unexploited:
‘/composition’ was generated by ‘/compose’.

Sentence Pair 12

URIs exploited:
`Derek illustrated chart 1.`

URIs unexploited:
`‘/derek’ generated ‘/chart1’ by ‘/illustrate’.`

Sentence Pair 13

URIs exploited:
`Derek did compose on behalf of chartgen.`

URIs unexploited:
`‘/derek’ did ‘/compose’ on behalf of ‘/chartgen’.`

Sentence Pair 14

URIs exploited:
`Derek illustrated chart 1 from composition.`

URIs unexploited:
`‘/derek’ derived ‘/chart1’ from ‘/composition’ by ‘/illustrate’.`

Sentence Pair 15

URIs exploited:
`2 agent posted ride requests 1.`

URIs unexploited:
`‘/rideshare/#/users/agent2’ generated ‘/rideshare/#/rideRequests/1’ by ‘/rideshare/#post_ride_request_100339’.`

Appendix C

Paragraph pairs

These are the paragraph pairs that were used in the human evaluation in Chapter 5. For each paragraph pair, there are two paragraphs that were generated from the same PROV graph. In each case, one paragraph was generated in a chronological order, whilst the other was generated in an anti-chronological order. This ordering was achieved using a topological sort.

Pair 1

Topological Sort

Store ride request 81935 used ride request 1.
Store ride request 81935 was informed by post ride request 100339.
Store ride request 81935 was associated with ride server.
Ride request 1 was generated by post ride request 100339.
Agent 2 posted ride request 1.
Post ride request 100339 was associated with agent 2.

Reverse Topological Sort

Post ride request 100339 was associated with agent 2.
Agent 2 posted ride request 1.
Ride request 1 was generated by post ride request 100339.
Store ride request 81935 was associated with ride server.
Store ride request 81935 was informed by post ride request 100339.
Store ride request 81935 used ride request 1.

Pair 2

Topological Sort

Store ride request 126023 was associated with ride server.
Store ride request 126023 was informed by post ride request 102205.
Store ride request 126023 used ride request 0.
Ride request 0 were generated by post ride request 102205.
Agent 1 posted ride request 0.
Post ride request 102205 was associated with agent 1.

Reverse Topological Sort

Post ride request 102205 was associated with agent 1.
Agent 1 posted ride request 0.
Ride request 0 were generated by post ride request 102205.
Store ride request 126023 used ride request 0.
Store ride request 126023 was informed by post ride request 102205.
Store ride request 126023 was associated with ride server.

Pair 3

Topological Sort

Star rating total was attributed to reputation manager.
Star rating average was attributed to reputation manager.
Star rating total was attributed to reputation manager.
Reputation manager generated opinion 2.
Reputation manager generated opinion 1.
Reputation manager generated reputation 2.
Reputation manager generated reputation 1.
Generate reputation 1 was informed by store record 1.
Generate reputation 1 was associated with reputation manager.
Generate opinion 1 was informed by store record 1.
Generate opinion 1 was associated with reputation manager.
Store record 1 was informed by post 123.
Store record 1 was associated with reputation manager.
Feedback 1 was attributed to reputation manager.
Rm posted application 1.
Post 123 was associated with rm.

Reverse Topological Sort

Post 123 was associated with rm.
Rm posted application 1.
Feedback 1 was attributed to reputation manager.
Store record 1 was associated with reputation manager.
Store record 1 was informed by post 123.
Generate opinion 1 was associated with reputation manager.
Generate opinion 1 was informed by store record 1.
Generate reputation 1 was associated with reputation manager.
Generate reputation 1 was informed by store record 1.
Reputation manager generated reputation 1.
Reputation manager generated reputation 2.
Reputation manager generated opinion 1.
Reputation manager generated opinion 2.
Star rating total was attributed to reputation manager.
Star rating average was attributed to reputation manager.
Star rating total was attributed to reputation manager.

Pair 4

Topological Sort

Reputation 7 was attributed to rs.
Reputation 7 was generated by computing reputation 0.
Computing reputation 0 used feedback 1.
Computing reputation 0 used feedback 2.
Copmuting reputation 0 was an activity that started at 2015-05-12T15:54:58.557000+01:00 and ended at 2015-05-12T15:54:58.557000+01:00.
Computing reputation 0 was informed by preprocessing 0.
Computing reputation 0 used feedback 3.
Computing reputation 0 used feedback 4.
Feedback 4 was attributed to rs.

Reverse Topological Sort

Feedback 4 was attributed to rs.
Computing reputation 0 used feedback 4.
Computing reputation 0 used feedback 3.
Computing reputation 0 was informed by preprocessing 0.

Copmuting reputation 0 was an activity that started at 2015-05-12T15:54:58.557000+01:00 and ended at 2015-05-12T15:54:58.557000+01:00.
Computing reputation 0 used feedback 2.
Computing reputation 0 used feedback 1.
Reputation 7 was generated by computing reputation 0.
Reputation 7 was attributed to rs.

Pair 5

Topological Sort

Response 1641 was generated by 3da87f79.
Response 1641 was attributed to rs.
Bc4feff used opinion 1001.
Bc4feff used reputation 1006.
Bc4feff used reputation 1005.
Bc4feff used opinion 999.
Bc4feff used reputation 1004.
Bc4feff used reputation 1002.
Bc4feff used opinion 1000.
Bc4feff used reputation 1003.
Bc4feff was associated with rs.
Bc4feff used opinion 997.
Bc4feff used opinion 998.
Bc4feff was an activity that started at 2015-07-20T16:23:23.749000+01:00 and ended at 2015-07-20T16:23:23.749000+01:00.
Bc4feff was informed by 3da87f79.
Bc4feff used request 1641.
Reputation 1006 was attributed to rs.
Reputation 1005 was attributed to rs.
Reputation 1004 was attributed to rs.
Reputation 1003 was attributed to rs.
Reputation 1002 was attributed to rs.
Opinion 999 was attributed to rs.
Opinion 998 was attributed to rs.
Opinion 997 was attributed to rs.
Opinion 1001 was attributed to rs.
Opinion 1000 was attributed to rs.

Reverse Topological Sort

Opinion 1000 was attributed to rs.
Opinion 1001 was attributed to rs.
Opinion 997 was attributed to rs.
Opinion 998 was attributed to rs.
Opinion 999 was attributed to rs.
Reputation 1002 was attributed to rs.
Reputation 1003 was attributed to rs.
Reputation 1004 was attributed to rs.
Reputation 1005 was attributed to rs.
Reputation 1006 was attributed to rs.
Bc4feff used request 1641.
Bc4feff was informed by 3da87f79.
Bc4feff was an activity that started at 2015-07-20T16:23:23.749000+01:00 and ended at 2015-07-20T16:23:23.749000+01:00.
Bc4feff used opinion 998.
Bc4feff used opinion 997.
Bc4feff was associated with rs.
Bc4feff used reputation 1003.
Bc4feff used opinion 1000.
Bc4feff used reputation 1002.
Bc4feff used reputation 1004.
Bc4feff used opinion 999.
Bc4feff used reputation 1005.
Bc4feff used reputation 1006.
Bc4feff used opinion 1001.
Response 1641 was attributed to rs.
Response 1641 was generated by 3da87f79.

Pair 6

Topological Sort

286 was attributed to rs.
286 was generated.
Generate opinion was an activity that started at 2015-01-10T10:01:25+00:00 and ended at 2015-01-10T10:01:25.100000+00:00.
Generate opinion was associated with rs.
Generate opinion was informed by store activity.

Rs was an agent.
138 was an entity.
137 was an entity.
136 was an entity.

Reverse Topological Sort

136 was an entity.
137 was an entity.
138 was an entity.
Rs was an agent.
Generate opinion was informed by store activity.
Generate opinion was associated with rs.
Generate opinion was an activity that started at 2015-01-10T10:01:25+00:00 and ended at 2015-01-10T10:01:25.100000+00:00.
286 was generated.
286 was attributed to rs.

Pair 7

Topological Sort

Agent switch api v 1 parsed api response disaggregation 1.
Result 1 was loaded at 2012-11-11T10:51:55.897000+00:00.
Matlab disaggregation loaded result 1.
Load disaggregation 1 was an activity that started at 2012-11-11T10:51:52.801000+00:00 and ended at 2012-11-11T10:51:55.897000+00:00.
Load disaggregation 1 used history consumption 1 at 2012-11-11T10:51:52.801000+00:00.
Load disaggregation 1 was associated with matlab disaggregation.
History consumption 1 was generated at 2012-11-11T10:51:52.794000+00:00 by serving external data 1.
Electric20 server served history consumption 1.
The serving external data 1 was started at 2012-11-11T10:51:51.340000+00:00 by 1.
Serving external data 1 was an activity that started at 2012-11-11T10:51:51.340000+00:00 and ended at 2012-11-11T10:51:52.800000+00:00.
Serving external data 1 was associated with 20 server electric.
Api request disaggregation 1 was an activity that started at 2012-11-11T10:51:52.801000+00:00 and ended at 2012-11-11T10:51:55.898000+00:00.
Api request disaggregation 1 was associated with agent switch api v 1.
Api request parsing 1 used http request 1 at 2012-11-11T10:51:51.340000+00:00.

Api request parsing 1 was an activity that started at 2012-11-11T10:51:51.339000+00:00 and ended at 2012-11-11T10:51:55.900000+00:00.
Api request parsing 1 was associated with agent switch api v 1.

Reverse Topological Sort

Api request parsing 1 was associated with agent switch api v 1.
Api request parsing 1 was an activity that started at 2012-11-11T10:51:51.339000+00:00 and ended at 2012-11-11T10:51:55.900000+00:00.
Api request parsing 1 used http request 1 at 2012-11-11T10:51:51.340000+00:00.
Api request disaggregation 1 was associated with agent switch api v 1.
Api request disaggregation 1 was an activity that started at 2012-11-11T10:51:52.801000+00:00 and ended at 2012-11-11T10:51:55.898000+00:00.
Serving external data 1 was associated with 20 server electric.
Serving external data 1 was an activity that started at 2012-11-11T10:51:51.340000+00:00 and ended at 2012-11-11T10:51:52.800000+00:00.
The serving external data 1 was started at 2012-11-11T10:51:51.340000+00:00 by 1.
Electric20 server served history consumption 1.
History consumption 1 was generated at 2012-11-11T10:51:52.794000+00:00 by serving external data 1.
Load disaggregation 1 was associated with matlab disaggregation.
Load disaggregation 1 used history consumption 1 at 2012-11-11T10:51:52.801000+00:00.
Load disaggregation 1 was an activity that started at 2012-11-11T10:51:52.801000+00:00 and ended at 2012-11-11T10:51:55.897000+00:00.
Matlab disaggregation loaded result 1.
Result 1 was loaded at 2012-11-11T10:51:55.897000+00:00.
Agent switch api v 1 parsed api response disaggregation 1.

Pair 8

Topological Sort

Derek compiled chart 1.
Chart 1 was compiled at 2012-03-02T10:30:00+00:00.
Derek illustrated chart 1 from composition.
Derek composed composition.
Article v 1 was an entity.
Compose used dataset 1.
Compose used region list.
Chart 2 was an entity.

Article v 2 was an entity.
Derek acted on behalf of chartgen.
Dataset 2 was generated by correct.
Blog entry was an entity.
Correct was an activity that started at 2012-03-31T09:21:00+01:00 and ended at 2012-04-01T15:21:00+01:00.
Correct used dataset 1.
Article was an entity.

Reverse Topological Sort

Article was an entity.
Correct used dataset 1.
Correct was an activity that started at 2012-03-31T09:21:00+01:00 and ended at 2012-04-01T15:21:00+01:00.
Blog entry was an entity.
Dataset 2 was generated by correct.
Derek acted on behalf of chartgen.
Article v 2 was an entity.
Chart 2 was an entity.
Compose used region list.
Compose used dataset 1.
Article v 1 was an entity.
Derek composed composition.
Derek illustrated chart 1 from composition.
Chart 1 was compiled at 2012-03-02T10:30:00+00:00.
Derek compiled chart 1.

Pair 9

Topological Sort

Weights graphic was generated at 2016-03-02T19:40:22+00:00 by matplotlib plot.
Matplotlib plot used weights.
Matplotlib plot used matplotlib.
Weights was generated at 2016-03-02T19:40:21+00:00 by read csv.
Read csv used weightReport-3-2-21-31-34-44.csv.
Read csv used pandas.
WeightReport-3-2-21-31-34-44.csv was attributed to user@googlemail.com.
WeightReport-3-2-21-31-34-44.csv was generated at 2016-03-02T20:31:34.867000+00:00

by create document.
Create document used weight db.
Create document was associated with weight companion.
Weight db was attributed to user@googlemail.com.

Reverse Topological Sort

Weight db was attributed to user@googlemail.com.
Create document was associated with weight companion.
Create document used weight db.
WeightReport-3-2-21-31-34-44.csv was generated at 2016-03-02T20:31:34.867000+00:00 by create document.
WeightReport-3-2-21-31-34-44.csv was attributed to user@googlemail.com.
Read csv used pandas.
Read csv used weightReport-3-2-21-31-34-44.csv.
Weights was generated at 2016-03-02T19:40:21+00:00 by read csv.
Matplotlib plot used matplotlib.
Matplotlib plot used weights.
Weights graphic was generated at 2016-03-02T19:40:22+00:00 by matplotlib plot.

Pair 10

Topological Sort

Target 9 2 was generated by 1411560570 812.
Target 9 2 was attributed to uav silver commander.
1411560570 812 was associated with uav silver commander.
1411560570 812 used target 9 1.
1411560570 812 was an activity that ended at 2014-09-24T13:09:30.812000+01:00.
Target 9 1 was attributed to crowd scanner.
Target 9 0 was attributed to crowd scanner.
Report 64 was attributed to crowdreporter 1.
Report 43 was attributed to crowdreporter 1.
Report 33 was attributed to crowdreporter 2.
Report 2 was attributed to crowdreporter 3.
Report 16 was attributed to crowdreporter 3.

Reverse Topological Sort

Report 16 was attributed to crowdreporter 3.
Report 2 was attributed to crowdreporter 3.
Report 33 was attributed to crowdreporter 2.
Report 43 was attributed to crowdreporter 1.
Report 64 was attributed to crowdreporter 1.
Target 9 0 was attributed to crowd scanner.
Target 9 1 was attributed to crowd scanner.
1411560570 812 was an activity that ended at 2014-09-24T13:09:30.812000+01:00.
1411560570 812 used target 9 1.
1411560570 812 was associated with uav silver commander.
Target 9 2 was attributed to uav silver commander.
Target 9 2 was generated by 1411560570 812.

Appendix D

Human explanations — Annotated transcripts

These are the annotated transcripts from the experiment in Section 5.1.

D.1 Participant 1

Example Graph

OK, so this is about John (1C) baking (7B) a cake (8A). And he used a bunch of ingredients (6D) which were sugar (2H) flour (3G) eggs (4F) and butter (5E). Yeah, he made a cake.

Graph 1

So, this is about... the creation of a ride request (4E). It was originated from agent 1 (1D), and it caused, a post ride request (2C) activity um which stored (5A) this ride request on a... by a ride server (3B), which was a ride server. Yep.

Graph 2

This seems to be about um a case (4G) that was generated by or raised (3H) by Alistair Hughes (1F), um about some patients (2I), um and then a process of investigation (5E) that began with a case document, and it generated an investigation (7D), and then that investigation seemed to involve something to do with parkinsons. And then the investigation seemed to be used (8B) by John Moorley (6C), and it looked as though it was looking for a specific gene. It's a variant (9A). I don't know what a variant

investigation versus an investigation is. And there's a bunch of other bits and pieces associated with each activity that don't make a great deal of sense.

Graph 3

So this is... it seems to be related to a weight app (2F). And I'm guessing it was, someone called user/onyame (1B) and they have a weight database (3G) and it was generating a document (4E) taking all the weight reports into a csv file (6D) that was then read by a library called pandas (5I) in python (7H) and it generated a, um, weights file (9C) which was read in by a plotting function in matplotlib (10J) (8K), and it generated probably a graphics of the weights (11A). So that was, the graphic was derived from a visualisation of the weights. It happened a couple of weeks ago.

Graph 4

So, this is about, er, a, I guess, user (1C) up here, I won't try to read out its identifier, erm, but it looks like they submitted a request (3E) (8B) possibly to view something (2D). The submission process then was received by something else on behalf of the user and then that generated a response based on a couple of things (6G): two ride requests (4I) (5H) and I don't know if that's the original request — (7F) three ride requests, sorry — and that was a response (9A), and that seemed to be all it did. There's some weirdness here, what does that mean? That was generated from ... some kind of view process generated a request. I guess that was then used by the sending request and that request was what the response was derived from as well as the ride requests

Graph 5

OK, so, there are a couple of things that are here, so it seems that there was a chart (17A) generated from a composition (11E) of two entities a dataset (2H) and a region list (3G). So the dataset looks like the idea was to aggregate values in the dataset by region and that was done by a compose process (8F). And it was started by Derek (4B), who belongs to an organisation called Chart Generators (1C). The aggregation process generated a composition that was illustrated (15D) to generate a chart. I don't know what the compile thing is (10I). It seems to be that the chart was compiled, by something. It doesn't necessarily say exactly what. Meanwhile the dataset was also used elsewhere... there was an activity that corrected (5O) the dataset to generate another dataset (6N) – a revision of it – and there's a chart 2 (12P) – i don't know if that's related to chart 1, probably not – that was derived from that and also there was a series of articles that... sorry, a series of versions (9M) (13L) (7K) of an article about crime rises in the city that were generated from the two datasets. It looks as though somebody

wrote an article about the first dataset, found a problem, and then corrected the dataset, and then updated their article, and they also wrote a blog entry (14J) about it... sorry, rather there was a blog entry that quoted part of the article. But it doesn't say which revision of the article.

Graph 6

Again, we've got a User 1 (1H) who was associated with, or instigated a post agreement (2G) with some strange parameters on it. As a result of that, there was a thing that updated some ride plans (6E), based on that submission (4F). That was performed by the ride server (3D). And as a result of updating the ride plans the new ride plan that was derived from the previous ride plan ... ok, there were two ride plans (7B) (8A), no idea why there's two – they look very similar, as far as their IDs are concerned – don't know what parp and arp mean, ride plan presumably. And there are some strange identifiers on the processes. There seems to be a distinction about whether things are the result of human input or as a result of machine output.

Graph 7

So this seems to be about three users: Joe (1M), Alice (2G), and Alex (7C). Alice and Joe posted (3L) (4F) some ride requests (5K) (6E) and those were used to generate a potential ride plan (8W), which Joe accepted (9X) and Alice accepted (11Q), so one of them seems to have been a rider (14P), and one a commuter (13R). So they agreed on the ride, and then along came Alex, who also posted a ride request (12I) (15H). And that was matched (17J) against the original ride requests and a new potential ride plan (22D) was created that invalidated the previous ride plan and then Alex seemed to reject the ride plan, twice (24B) (23A), which is slightly odd, at the same time. And then there's a third ride plan (21N) that presumably involves all three ride plans: one, two, three.

Graph 8

This is about an API (2H), a Web API that submitted a request (1I). The API request started two processes (3G): one something to do with disaggregation (6K), and the other was about extracting some external data (5E) from another server. That server (4F) generated a history consumption document (8D) – presumably some data. Meanwhile, the disaggregation request triggered a load disaggregation (9C)into matlab, a matlab process (7J) and that used the history consumption and it generated a disaggregation result (10B) that was then used as a part of an API response (11A), presumably to the

original request. It looks like it was about 6 minutes of data, but that's about all of interest, I think.

Graph 9

So this seems to be about user 0 (1G) posting an agreement (2F), he started a post agreement process that generated an answer document (4E). As a result of that, an update ride plan (6D) was started by the ride server (3C). It used the response the user made, and it generated something called a ride plan (7A), that was derived from an original ride plan. (5B)

Graph 10

So this is about an activity or something called rep (1C). It received a request (2E), and the process that received that generated a Request Response document (3D). That document was then used by an activity that computed reputation, computing a response (4F). It generated a response document (6A), and an activity of sending the response (5B) *** inaudible mumbling ***. But yeah, it looks as if the third process was sending the response back to somewhere, presumably external to the reputation service. There's some weirdness in this graph that I don't quite follow.

D.2 Participant 2

Example Graph

So, John (1C) baked (7B) a cake (8A) using the ingredients (6D) sugar (2H), flour (3G), eggs (4F), and butter (5E).

Graph 1

So, I would start off with the agent, so I guess to describe this graph I would say: An agent (1D) who was logged in posted (2C) a ride request and that ride request (4E) was stored (5A) by the ride server (3B).

Graph 2

So, a user, Alistair Hughes (1F), created (3H) a case (4G) and an (5E) investigation (7D) based on that case, which was about patient (2I). That investigation was used by

a variant investigation (8B) by another user John Morley (6C), and that resulted in a variant (9A).

Graph 3

So, a person (1B) submitted their weight (3G) and a software agent (2F) created a document (4E) (6D) reporting their weight. That weight was then processed using a python method (7H) to read that data and plot the graph (10J) (11A). That generated various entities describing weights (9C) and also it describes the libraries (5I) (8K) which it used.

Graph 4

So, a user (1C) sees a view (2D) and that view... they send a request (3E) (8B) and that request has a response (9A) and that response consists of two ride requests (4I) (5H). I think it's worth noting that the ride requests are versioned.

Graph 5

So, Derek (4B), on behalf of an organisation which generates charts (1C), illustrates (15D) a chart (17A) and he used a composition (11E) entity and various other datasets (2H) (6N) derived from that which is somehow associated with a blog entry. (14J)

Graph 6

So a user (1H) posted an agreement (2G), and that agreement (4F) is used by a ride server to generate (6E) an agreed ride plan (8A), which was derived from a potentially agreed ride plan (7B), and, that was derived from , by the looks of it, an original ride plan (5C) that was attributed to the ride server. (3D)

Graph 7

So, I would start by saying that there are three users: Joe, Alice, and Alex , all of which submit ride request sand those ride requests are used to generate potential ride plans . Start again: I'd say that Joe (1M), Alice (2G), and Alex (7C) posted ride requests (3L) (5K) (4F) (6E) (12I) (15H) and the ride server (10S) matched those requests and created ride plans (8W) (22D) (14P). And then I'd go on to talk about who accepted and it looks like Alice accepted a potential ride plan (11Q), and so did Joe (9X), and Alex rejected the second ride plan which means that then there's one ride that happened (23A). That

one's more complicated. I'd like to take an overview of it, rather than going into the individual entities, and what exactly happened, but trying to describe the processes and some of the more interesting features by saying that one ride plan was agreed in this transaction.

Graph 8

So I would start by saying that there's a switch agent (2H) and it receives an HTTP request (1I), and it's parsed (3G)and that starts, actually I'm unsure what that means - the light grey arrows. I've not used that. Theres a connection between the other two methods (6K) (5E), and it looks like they were started with the processing of that request and two servers (4F). One server records the history of the consumption (8D) of that data, and that data is processed using a disaggregation matlab (9C) (7J) thingy and creates some disaggregations (10B) from that API result(11A). To read that graph, I would prefer that there were types. I would find that more useful. I find that quite confusing.

Graph 9

So, there's an agent (1G) who posts an agreement (2F) on a ride plan and that agreement (4E) is used by the ride server (3C) to generate (6D) a potentially agreed ride (7A), which is derived from the original ride plan (5B). The potentially agreed ride plan is a version of that original ride.

Graph 10

A reputation peer (1C) receives a ride (2E) request and then computes (4F) a response, and then sends a (5B) response and that response (6A) is derived from a reputation report (3D).

D.3 Participant 3

Example Graph

So, right, there is a cake (8A), which here is called ex:cake. It was baked (7B) by John (1C), and it was baked from a number of ingredients (6D) including sugar (2H), flour (3G), eggs (4F), and butter (5E).

Graph 1

So, this is an action store ride request (5A) number 126023. It was done by the ride server (3B) in response to a ride request (4E), which was the result of the action post ride request (2C) initiated by user agent 1 (1D).

Graph 2

So, this is the descriptions of a variant (9A) of genes, I think. It was identified in action to investigate the variant (8B), carried out by a user named Jonny Morley (6C). This was initiated by... as part of an investigation (7D) called demo. So the investigation was created (5E) as part of a case (4G) concerning a patient (2I), and it was created (3H) by a user called Alistair Hughes (1F).

Graph 3

So, this shows there's a plot (11A) about a person's weight that was generated (10J) by a library called matplotlib version 1.5.1 (8K). It used the weight data (9C) that was originally read (7H) from a csv file (6D) called WeightReport-3-2-21-31.34.44.csv. So this weight csv file was created (4E) by an agent called app weight companion (2F), and it was created from the weight database (3G) attributed to user onyame@googlemail.com (1B).

Graph 4

So, this shows the response (9A) that was generated from a number of ride requests (4I) (5H) (7F). Essentially, it's a response to a service request (8B) that was sent by a user (3E) (1C), identified by user_9f9d. I think that's all the relevant information in this graph.

Graph 5

So, this graph shows essentially two different processes. The first one, it was about a blog entry (14J) that quoted an article (7K). This article went through two separate revisions (9M) (13L). The first one used a dataset called ex:dataset1 (2H), and version 2 used a dataset ex:dataset2 (6N), which was a corrected (5O) version of ex:dataset1. The other process is about a chart called ex:chart1 (17A) that was created by a person called Derek (4B) in order to illustrate a composition (15D) (11E) of the dataset1 and another data called ex:regionlist (3G). Derek is working on behalf of a company called Chart Generators Inc. (1C)

Graph 6

So, this graph shows a ride plan (8A) identified by ridePlans/0/v/arp. It was generated by the ride server in (6E) (3D) response to a ride agreement (4F) posted by (2G) rs:/users/1 (1H) and I guess, this response was essentially it agreed.

Graph 7

So, this graph shows the interactions of three users name Joe (1M), Alice (2G), and Alex (7C), with the ride server (10S). These three users post ride request (3L) (5K) (4F) (6E) (12I) (15H) to the server which generated (16O) potential ride plans according to those requests (8W) (22D) (21N). There are three different potentially agreed ride plans generated. The first two were rejected by user Alex (23A) (24B), which leads to the last one. It doesn't show if that was rejected or not. But essentially, I'd assume that this new potentially agreed ride plan was generated after the rejection by Alex.

Graph 8

This graph explains how an API response (11A) was produced in response to HTTPRequest_1 (1I). So the API response used this aggregation result (10B) that was generated by a user historic electrical consumption (8D) and the disaggregation (9C) was done by an agent called MatlabDisaggregation (7J). The HistoryConsumption_1 here was retrieved from an external server called as:Electric20Server (4F), as part of the response to the HTTP request.

Graph 9

This graph shows a ride plan (7A) that was updated (6D) based on the answers (4E) posted (2F) by a user identified by 0 (1G). The ride plan (5B) was originally created by the ride server. (3C)

Graph 10

This graph shows a response (6A) generated by an agent called rep:rs/ (1C) which is a reputation peer. It generated a response based on a reputation identified by rep:rs/application/9/reputation/1078. (3D)

D.4 Participant 4

Example Graph

So, my understanding about this graph. Here I see four ingredients (6D), they are a part of ingredients. Ingredients is a collection of many ingredients which are sugar (2H), flour (3G), eggs (4F), and butter (5E). Based on these ingredients, a cake (8A) was created which was made by the actor John (1C), through the process of baking (7B). So, here I see was associated with, and there is a graph from baking to John indicating that John baked the cake, sorry John baked the ingredients to produce the cake, and the ingredients contain sugar, flour, eggs, and butter.

Graph 1

So here I see the process of post_ride_request_102205 (2C), it was used by agent1 (1D) to produce rideRequests/0 (4E). Later the entity rightRequests/0 is used by store_ride_request126023 (5A). And that store_ride_request126023 is used by the actor ride_server (3B). Here I see the arrow from store_ride_request126023 towards post_ride_request_102205, indicating that the post_ride_request_102205 started before the store_ride_request126023 so it was inheritance by the post_ride_request_102205.

Graph 2

Here I see a process of a case created (3H). It used entity patients (2I) to create an entity case (4G), and the process of case_created is processed by user1 (1F). Later after the entity case... sorry, the name of user1 is Alistair Hughes. And after the case entity was created, it was later used by process investigation_created (5E), also performed by user Alistair Hughes, to create the investigation entity (7D). After the investigation entity was created, it was then used by the process of variant_investigation (8B) performed by Jonny Morley (6C) to create an entity variant (9A).

Graph 3

Here I see a process of create_document (4E) performed by WeightCompanion (2F) actor here, the agent WeightCompanion. It used the userdata:weight_db (3G), it belongs to user/onyame@googlemail.com (1B). After the process of create_document, it created userdata:WeightReport-3-2-21-31.34.44.csv (6D) and later this entity was used together with pandas library (5I) by a process python_method:read_csv (7H). After the activity of python_method:read_csv, it generated userdata:weights (9C), and here I see the arrow between userdata:weights and userdata:WeightReport-3-2-21-31.34.44.csv, indicating that

userdata:weights was derived by the userdata:WeightReport-3-2-21-31.34.44.csv. Later userdata:weights was used by, together with matplotlib library (8K), by python_method: matplotlib_plot (10J) to generate graphic/weights (11A) and graphic/weights also belongs to the first user user/onyame@googlemail.com.

Graph 4

In this graph, I only see one user (1C). It's a random user number, and it has an entity with a random name with type cas:View (2D). This entity was used by the activity (3E) with type cas:sending_request to generate an entity with type cas:request (8B) indicated by 8 in picture here. Entity 8 also belongs to the actor with the random number with type Peer, and after the first process, it goes through the second process (6G), also with a random number associated with first user and generated the random number type cas:Response. The final result is an entity with a random number and type cas:Response (9A). It derived also from four entities 447/v/0 544/v/2 448/v/3 and 8. (4I) (5H) (7F) That's what I can see about this graph.

Graph 5

I see here an agent which has a type organisation (1C), and the organisation is called Chart Generators Inc. There is also an agent which has the type Person, and his name is Derek (4B). Derek acts on behalf of this company (Chart Generators Inc.). Derek was associated with the activity of compose (8F), and compose used two inputs regionList (3G) and dataset1 (2H) in order to generate the entity composition (11E). The person who performed the activity is Derek, and he acts on behalf of his company, Chart Generators. If you look at dataset1, it was also used by the activity correct (5O) to generate dataset2 (6N), so dataset2 was derived from dataset1. After dataset2 was created it also created chart2 (12P), so we can say that chart2 is derived from dataset2. And articleV1 (9M) also derived from dataset1 and articleV1 is a specialisation of article. Article (7K) here, has the title "Crime Rises in Cities", and I also see articleV2 (13L) which was also an alternative of articleV1. I see a blog entry (14J) was derived from article. So here we can say that articlev1 and articlev2 are both of the type article. And I also see entity chart1 (17A) that was created by the process of illustrate (15D), and the agent that performed the activity illustrate was also Derek. Chart 1 was actually created by the process that is called compile (10I) and I also see compile2, (16Q) though there are no arrows coming in or going out of that activity. It's by itself.

Graph 6

I see two activities post_agreement (2G) and update ride_plans (6E). For the first activity, post_agreement, it was performed by User1 (1H), and it generated an entity answer84923 (4F), and activity number 2 update_rideplans was performed after activity number 1 has finished, and it generated first ridePlans/0/v/parp (7B), with type MachineOutput, and after it generated MachineOutput it later generated ridePlans/0/v/arp, (8A) also of the same type, MachineOutput. The second activity which is update_rideplans was performed by ride_server (3D) and the agent ride_server also ahas an entity ridePlans/0. (5C) This entity was a source of an entity that was produced by the second activity.

Graph 7

I see four agents here, Joe (1M), Alice (2G), Alex (7C), and ride_server (10S), and I see Joe, Alice, and Alex as a user who'd like to request to have a ride request. So I see post_ride_request_1 (4F) which belongs to Alice, and the activity post_ride_request_2 (3L) was associated with agent or user Joe, and post_ride_request_3 (12I) was associated with user Alex. All of them requested to have a ride request and each of them produced an entity that belongs to their own request. So Joe has joe_ride_request (5K), Alice has alice_ride_request (6E), and Alex also has alex_ride_request (15H). Later there is an activity of joe_accept_potential_ride_plan (9X) that used the entity of potential_ride_plan_1 (8W) that was derived from joe_ride_request. This indicates that someone has accepted the ride request from Joe. Also from Alice. *** INAUDIBLE MUMBLE *** I also see there is around two activities of accept_potential_ride_plan (11Q), and the users that are associated with those two activities are Joe and Alice, so my understanding is that maybe Joe and Alice had some time to do a ride together. And Alex, Alex here... I see two activities alex_reject_ride_plan_2 (23A) and alex_reject_ride_plan_1 (24B). Maybe Alex rejected the two plans that used potential_ride_plan_2, but this is difficult for me to understand.

Graph 8

I see the activity of APIRRequestParsing_1 (3G) performed by agentSwitchAPIv1 (2H), and this process APIRequestParsing_1 used an entity of HTTPRequest_1 (1I). After this activity, it goes to activity ServingExternalData_1 (5E) performed by Electric20Server (4F) because I see the starting time. The starting time of activity ServingExternalData_1 is earlier than APIRequestDisaggregation_1 (6K). So activity of ServingExternalData_1 produced HistoryConsumption_1 (8D); it was an entity, and it was used by LoadDisaggregation_1 (9C) performed by MatlabDisaggregation (7J). The activity of loadDisaggregation_1 is started after the activity APIRequestDisaggregation_1 was finished. And the activity of loadDisaggregation_1 generated DisaggregationResult_1 (10B)

and I see the entity of APIResponseDisaggregation_1 (11A) and it was derived based on the entity DisaggregationResult_1.

Graph 9

I see two activities post_agreement (2F) and update_rideplans (6D). The activity of post_agreement was associated with users/0 (1G) and it, post_agreement, generated answer108117 (4E). After entity answer108117 was generated it was then used by the second activity update_rideplans, and update_rideplans is started after the first activity, post_agreement, finished. The second activity, update_rideplans, is performed by ride_server (3C), another agent, and it generated — the second activity, update_rideplans generated ridePlans/0/v/parp (7A), and it has a type MachineOutput. This MachineOutput was derived from ridePlans/0 (5B), also another entity, and that entity belongs to the ride_server, which is a Webserver.

Graph 10

Here I see three consecutive activities (2E) (4F) (5B), all of them with random numbers. Activity 1 pre_1 it was performed by the only agent (1C) in the graph the RepuationPeer. It then generated an entity called rs/application/9/reputation/1078/ (3D). This entity was then used by the second activity rep:/rs random number to produce the final entity, which is rs/response/2699 (6A). Then after that, the third activity, also performed by ReputationPeer used the same entity as the activity 2 used before, and it also produced the same result as activity 2 produced before.

D.5 Participant 5

Example Graph

Ok this graph is about describing baking (7B) a cake (8A) by john (1C) using four ingredients (6D) which are sugar (2H), flour (3G), eggs (4F), and butter (5E).

Graph 1

The ride server agent (3B) used (5A) a ride request (4E) that had been generated by agent1 (1D). It had been generated by an activity (2C) which was associated with agent1.

Graph 2

User1 (1F) started an investigation (5E) (7D) based on a case (4G) which had been generated (3H) from Patients (2I) information. This investigation is used (8B) by User (another user) (6C) to generate a variant. (9A)

Graph 3

This graphic/weights (11A) has been generated (10J) from weights (9C) and matplotlib library (8K). The weights comes (7H) from the WeightReport (6D) and a library called pandas (5I). The weight report has been derived (4E) from a weight database (3G) which is attributed to a user with email onyame@googlemail.com (1B).

Graph 4

A ride request has been generated from different, ... ride response (9A), sorry, a ride response has been generated or derived from different ride requests (4I) (5H) (7F). Somehow these ride requests come from an agent. (1C)

Graph 5

A chart (17A) was generated by an agent Derek (4B) who acted on behalf of another agent chartgen (1C). The chart is derived from, or a dataset (2H) has been used in deriving the chart1, the dataset itself has been derived from another dataset (6N)or many articles (9M)(13L)(7K). The articles represent information about Crimes in Cities.

Graph 6

A sequence of ride plans (5C) (7B) (8A) have been generated (6E) by a ride_server (3D) agent using a set of answers (4F) generated (2G) as part of an agreement generated by the agent users/1. (1H)

Graph 7

A set of potential ride plans have been generated from many ride riquest made by agents, Alex, Alice, and Joe, using different plans, and previously agreed ride plans.

Graph 8

An APIResponseDisaggregation (11A), library or class, has been generated using DisaggregationResult (10B) that come from MatlabDisaggregation (7J) using HistoryConsumption (8D) that was generated under the control of Electric20Server (4F).

Graph 9

A ridePlan (7A) was derived from another ridePlan (5B) which was attributed to a Webserver (ride_server) (3C) based on a set of answers (4E) which came from an agreement (2F) made under the control of users/0. (1G)

Graph 10

A ride share response (6A) has been generated from a reputation report (3D) created by a ride share application (1C), under the control of the agent rs (a reputation peer).

References

Adida, B., Birbeck, M., McCarron, S., and Hermann, I. (2015). RDFa Core 1.1 – Third Edition. Recommendation of the *World Wide Web Consortium*, `https://www.w3.org/TR/rdfa-core/`.

Basile, V. (2016). A repository of frame instance lexicalizations for generation. In *Proceedings of the 2nd International Workshop on Natural Language Generation and the Semantic Web*.

Bateman, J. and Zock, M. (2003). Natural Language Generation. In Mitkov, R., editor, *The Oxford Handbook of Computational Linguistics*, pages 284–304. Oxford University Press, Oxford.

Beckett, D. (2014). RDF 1.1 N-Triples. Recommendation of the *World Wide Web Consortium*, `https://www.w3.org/TR/n-triples/`.

Beckett, D., Berners-Lee, T., Prud'hommeaux, E., and Carothers, G. (2014). RDF 1.1 Turtle. Recommendation of the *World Wide Web Consortium*, `https://www.w3.org/TR/turtle/`.

Berners-Lee, T. (1994). RFC 1630: Universal Resource Identifiers in WWW. Request for Comment of the *Internet Engineering Task Force* `https://www.ietf.org/rfc/rfc1630.txt`.

Berners-Lee, T. (1996). Universal Resource Identifiers – Axioms of Web Architecture. Design note, World Wide Web Consortium, `https://www.w3.org/DesignIssues/Axioms.html`.

Berners-Lee, T. (1998). Principles of Design. Design note, World Wide Web Consortium, `https://www.w3.org/DesignIssues/Principles.html`.

Berners-Lee, T. (1999). *Weaving the Web*. The Past, Present and Future of the World Wide Web by its Inventor. Orion Business, London, UK.

Berners-Lee, T. (2006). Linked Data. Design note, World Wide Web Consortium, `http://www.w3.org/DesignIssues/LinkedData.html`.

Berners-Lee, T. and Connolly, D. (2011). Notation3 (N3): A readable RDF syntax. Team Submission of the *World Wide Web Consortium*, https://www.w3.org/TeamSubmission/n3/.

Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The Semantic Web. *Scientific American*, 284(5):28–37.

Bernstein, A. and Kaufmann, E. (2006). Gino-a guided input natural language ontology editor. In *International Semantic Web Conference*, volume 2006, pages 144–157. Springer.

Bird, S., Klein, E., and Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media Inc, Sebastopol, CA.

Bizer, C. and Cyganiak, R. (2014). RDF 1.1 TriG. Recommendation of the *World Wide Web Consortium*, https://www.w3.org/TR/trig/.

Bontcheva, K. (2005). Generating tailored textual summaries from ontologies. *The Semantic Web: Research and Applications*, pages 531–545.

Bouayad-Agha, N., Casamayor, G., and Wanner, L. (2014). Natural language generation in the context of the semantic web. *Semantic Web*, 5(6):493–513.

Bouttaz, T., Eckhardt, A., Mellish, C., and Edwards, P. (2012). Integrating Text and Graphics to Present Provenance Information. In *Provenance and Annotation of Data and Processes Fourth International Workshop - IPAW*, pages 223–225. https://doi.org/10.1007/978-3-642-34222-6_20.

Bouttaz, T., Pignotti, E., Mellish, C., and Edwards, P. (2011). A policy-based approach to context dependent natural language generation. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 151–157. Association for Computational Linguistics.

Braines, D., Mott, D., Laws, S., de Mel, G., and Pham, T. (2013). Controlled english to facilitate human/machine analytical processing. *SPIE Defense, Security, and Sensing*, pages 875808–875808.

Brickly, D. and Miller, L. (2014). FOAF Vocabulary Specification 0.99. Published specification: http://xmlns.com/foaf/spec/.

Brin, S. (1999). *Extracting Patterns and Relations from the World Wide Web*, pages 172–183. Springer Berlin Heidelberg, Berlin, Heidelberg.

Buneman, P., Khanna, S., and Tan, W.-C. (2001). Why and Where: A Characterization of Data Provenance. In *International Conference on Database Theory*, pages 316–330. https://doi.org/10.1007/3-540-44503-X_20.

Burgess, L. C., Crotty, D., De Roure, D., Gibbons, J., Goble, C., Missier, P., Mortier, R., Nichols, T. E., and O'Beirne, R. (2016). Alan Turing Institute Symposium on Reproducibility For Data-Intensive Research — Final Report. Technical report.

Cahill, L., Carroll, J., Evans, R., Paiva, D., Power, R., Scott, D., and van Deemter, K. (2001). From RAGS to RICHES: exploiting the potential of a flexible generation architecture. In *th Annual Meeting of the Association for Computational Linguistics*, pages 106–113, Toulouse, France. https://doi.org/10.3115/1073012.1073027.

Cahill, L., Doran, C., Evans, R., Mellish, C., Paiva, D., Reape, M., Scott, D., and Tipper, N. (1999). In search of a reference architecture for NLG systems. In *7th European Workshop on Natural Language Generation*, pages 1–9.

Cahill, L., Doran, C., Evans, R., Mellish, C., Paiva, D., Reape, M., Scott, D., and Tipper, N. (2000). Reinterpretation of an existing NLG system in a Generic Generation Architecture. In *First international conference on Natural language generation*, pages 1–8.

Cheney, J., Missier, P., and Moreau, L. (2013). Constraints of the PROV Data Model. Recommendation of the *World Wide Web Consortium*, http://www.w3.org/TR/prov-constraints/.

Colin, E., Gardent, C., Mrabet, Y., Narayan, S., and Perez-Beltrachini, L. (2016). The webnlg challenge: Generating text from dbpedia data. In *Proceedings of the 9th International Natural Language Generation conference*, pages 163–167.

Corsar, D., Markovic, M., and Edwards, P. (2016). Social Media Data in Research: Provenance Challenges. In Mattoso, M. and Glavic, B., editors, *International Provenance and Annotation Workshop*, pages 1–4, McLean, Virginia, USA. Springer International Publishing. https://doi.org/10.1007/978-3-319-40593-3_20.

Cuevas-Vicenttin, V., Ludäscher, B., Missier, P., Belhajjame, K., Chirigati, F., Wei, Y., Dey, S., Koop, D., and Altintas, I. (2014). ProvONE: A PROV Extension Data Model for Scientific Workflow Provenance. Unofficial draft specification, http://vcvcomputing.com/provone/provone.html.

Cyganiak, R., Wood, D., and Lanthaler, M. (2014). RDF 1.1 Concepts and Abstract Syntax. Recommendation of the *World Wide Web Consortium*, https://www.w3.org/TR/rdf11-concepts/.

Davis, I., Steiner, T., and Le Hors, A. J. (2013). RDF 1.1 JSON Alternate Serialization (RDF/JSON). Working group note of the *World Wide Web Consortium*, https://www.w3.org/TR/rdf-json/.

DCMI Usage Board (2012). Dublin Core: DCMI Metadata Terms. Published specification of the *Dublin Core Metadata Initiative*, http://dublincore.org/documents/dcmi-terms/.

De Roure, D., Goble, C., and Stevens, R. (2007). Designing the myExperiment Virtual Research Framework for the Social Sharing of Workflows. In *Third IEEE International Conference on e-Science and Grid Computing (e-Science 2007)*, pages 179–186. IEEE. https://doi.org/10.1109/E-SCIENCE.2007.29.

Dragan, L., Luczak-Roesch, M., Berendt, B., Simperl, E., Packer, H., and Moreau, L. (2014). A-posteriori provenance-enabled linking of publications and datasets via crowdsourcing. In *2nd Workshop on Linking and Contextualizing Publications and Datasets*, London, UK. https://doi.org/10.1045/january2015-dragan.

Ell, B. and Harth, A. (2014). A language-independent method for the extraction of RDF verbalization templates. In *th International Natural Language Generation Conference*. Association for Computational Linguistics. http://anthology.aclweb.org/W/W14/W14-44.pdf#page=34.

Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures*. University of California, Irvine Doctoral dissertation.

Fuchs, N., Kaljurand, K., and Kuhn, T. (2008). Attempto controlled english for knowledge representation. *Reasoning Web*, pages 104–124.

Gandon, F., Schreiber, G., and Beckett, D. (2014). RDF 1.1 XML Syntax. Recommendation of the *World Wide Web Consortium*, https://www.w3.org/TR/rdf-syntax-grammar/.

Gatt, A. and Reiter, E. (2009). SimpleNLG: A realisation engine for practical applications. In *12th European Workshop on Natural Language Generation*, pages 90–93, Athens, Greece. http://dl.acm.org/citation.cfm?id=1610208.

Harris, S., Seaborne, A., and Prud'hommeaux, E. (2013). SPARQL 1.1 Query Language. Recommendation of the *World Wide Web Consortium*, https://www.w3.org/TR/sparql11-query/.

Hartig, O. and Zhao, J. (2009). Using Web Data Provenance for Quality Assessment. In *International Workshop on the role of Semantic Web in Provenance Management*. http://ceur-ws.org/Vol-526/paper_1.pdf.

Hartig, O. and Zhao, J. (2010). Publishing and Consuming Provenance Metadata on the Web of Linked Data. In *3rd International Provenance and Annotation Workshop*, Troy, NY, USA. https://doi.org/10.1007/978-3-642-17819-1_10.

Hasan, R., Sion, R., and Winslett, M. (2007). Introducing Secure Provenance: Problems and Challenges. In *2007 ACM workshop on Storage security and survivability*, pages 13–18, Alexandria, VA. https://doi.org/10.1145/1314313.1314318.

Hielkema, F. (2010). *Using natural language generation to provide access to semantic metadata*. PhD thesis, University of Aberdeen.

Hielkema, F., Mellish, C., and Edwards, P. (2007). Using wysiwym to create an open-ended interface for the semantic grid. In *Proceedings of the Eleventh European Workshop on Natural Language Generation*, pages 69–72. Association for Computational Linguistics.

Hielkema, F., Mellish, C., and Edwards, P. (2008). Evaluating an ontology-driven wysiwym interface. In *Proceedings of the Fifth International Natural Language Generation Conference*, pages 138–146. Association for Computational Linguistics.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hoekstra, R. and Groth, P. (2015). PROV-O-Viz - Understanding the Role of Activities in Provenance. In Ludäscher, B. and Plale, B., editors, *Provenance and Annotation of Data and Processes*. https://doi.org/10.1007/978-3-319-16462-5_18.

Hovy, E. (1987). Generating Natural Language Under Pragmatic Constraints. *Journal of Pragmatics*, 11:689–719. https://doi.org/10.1016/0378-2166(87)90109-3.

IBM Software (2014). The top five ways to get started with big data. Technical report.

Johnson, D. S. (1973). Approximation Algorithms for Combinatorial Problems. In *Fifth Annual ACM Symposium on Theory of Computing*, pages 38–49, Austin, TX. ACM. https://doi.org/10.1016/S0022-0000(74)80044-9.

Johnson, J., Karpathy, A., and Fei-Fei, L. (2016). Densecap: Fully convolutional localization networks for dense captioning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Karp, R. M. (1972). Reducibility among combinatorial problems. In Miller, R. E., Thatcher, J. W., and Bohlinger, J. D., editors, *Complexity of Computer Computations*, pages 85–103. 10.1007/978-1-4684-2001-2_9.

Klyne, G., Carroll, J. J., and McBride, B. (2004). Resource Description Framework (RDF): Concepts and Abstract Syntax. Recommendation of the *World Wide Web Consortium*, https://www.w3.org/TR/2004/REC-rdf-concepts-20040210/.

Kuhn, T. (2008). Acewiki: A natural and expressive semantic wiki. In *Fifth International Workshop on Semantic Web User Interaction (SWUI 2008)*.

Lebo, T., Sahoo, S. S., and McGuinness, D. L. (2013). PROV-O: The PROV Ontology. Recommendation of the *World Wide Web Consortium*, http://www.w3.org/TR/prov-o/.

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., and Bizer, C. (2015). DBpedia – A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*, 6(2):167–195. https://doi.org/10.3233/SW-140134.

Lester, J. C. and Porter, B. W. (1997). Developing and Empirically Evaluating Robust Explanation Generators: The KNIGHT Experiments. *Computational Linguistics*, 23(1):65–101. http://dl.acm.org/citation.cfm?id=972688.

Mann, W. C. and Thompson, S. A. (1988). Rhetorical Structure Theory: Toward a functional theory of text organization. *Text*, 8(3):243–281. https://doi.org/10.1515/text.1.1988.8.3.243.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330. http://dl.acm.org/citation.cfm?id=972475.

Markovic, M., Edwards, P., and Corsar, D. (2014). SC-PROV: A Provenance Vocabulary for Social Computation. In Ludäscher, B. and Plale, B., editors, *Provenance and Annotation of Data and Processes: 5th International Provenance and Annotation Workshop, IPAW 2014, Cologne, Germany, June 9-13, 2014. Revised Selected Papers*, pages 285–287. Springer International Publishing. https://doi.org/10.1007/978-3-319-16462-5_35.

McCrae, J., Spohr, D., and Cimiano, P. (2011). Linking Lexical Resources and Ontologies on the Semantic Web with Lemon. In *The Semantic Web Research and Applications*, pages 245–259. Lecture Notes on Computer Science. https://doi.org/10.1007/978-3-642-21034-1_17.

McNutt, M. (2015). Data, eternal. *Science*, 347(6217):7–7. https://doi.org/10.1126/science.aaa5057.

Mellish, C. and Dale, R. (1998). Evaluation in the context of natural language generation. *Computer Speech & Language*, 12(4):349–373. https://doi.org/10.1006/csla.1998.0106.

Mellish, C., Scott, D., Cahill, L., Paiva, D., Evans, R., and Reape, M. (2006). A Reference Architecture for Natural Language Generation Systems. *Natural Language Engineering*, 12(01):1–35. https://doi.org/10.1017/S1351324906004104.

Mellish, C. and Sun, X. (2005). Natural Language Directed Inference in the Presentation of Ontologies. In *10th European Workshop on Natural Language Generation*, Aberdeen, UK.

Mellish, C. and Sun, X. (2006). The semantic web as a Linguistic resource: Opportunities for natural language generation. *Knowledge-Based Systems*, 19(5):298–303. https://doi.org/10.1016/j.knosys.2005.11.011.

Mendel, T. (2008). *Freedom of Information: A Comparative Legal Survey*. UNESCO, Paris, 2nd edition.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Miller, G. A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41. https://doi.org/10.1145/219717.219748.

Missier, P., Dey, S., Belhajjame, K., Cuevas-Vicenttin, V., and Ludäscher, B. (2013). D-PROV: Extending the PROV Provenance Model with Workflow Structure. In *5th USENIX Workshop on the Theory and Practice of Provenance*, Lombard, IL, USA.

Moreau, L. (2010). The Foundations for Provenance on the Web. *Foundations and Trends in Web Science*, 2(2-3):99–241. https://doi.org/10.1561/1800000010.

Moreau, L. (2013). Food Supply Chain and Provenance. Personal blog, https://lucmoreau.wordpress.com/2013/08/06/food-supply-chain-and-provenance/.

Moreau, L. (2014). Aggregation by Provenance Types: A Technique for Summarising Provenance Graphs. In *Graphs as Models 2015*, pages 129–144. https://doi.org/10.4204/EPTCS.181.9.

Moreau, L., Clifford, B., Freire, J., Futrelle, J., Gil, Y., Groth, P., Kwasnikowska, N., Miles, S., Missier, P., Myers, J., Plale, B., Simmhan, Y., Stephan, E., and Van den Bussche, J. (2011). The Open Provenance Model Core Specification (v1.1). *Future generation computer systems*, 27(6):743–756. https://doi.org/10.1016/j.future.2010.07.005.

Moreau, L., Ludäscher, B., Altintas, I., Barga, R. S., Bowers, S., Callahan, S., Chin, G., Clifford, B., Cohen, S., Cohen-Boulakia, S., Davidson, S., Deelman, E., Digiampietri, L., Foster, I., Freire, J., Frew, J., Futrelle, J., Gibson, T., Gil, Y., Goble, C., Golbeck, J., Groth, P., Holland, D. A., Jiang, S., Kim, J., Koop, D., Krenek, A., McPhillips, T., Mehta, G., Miles, S., Metzger, D., Munroe, S., Myers, J., Plale, B., Podhorszki, N., Ratnakar, V., Santos, E., Scheidegger, C., Schuchardt, K., Seltzer, M., Simmhan, Y. L., Silva, C., Slaughter, P., Stephan, E., Stevens, R., Turi, D., Vo, H., Wilde, M., Zhao, J., and Zhao, Y. (2008a). Special Issue: The First Provenance Challenge. *Concurrency and Computation: Practice and Experience*, 20(5):409–418. https://doi.org/10.1002/cpe.1233.

Moreau, L. and Missier, P. (2013a). PROV-DM: The PROV Data Model. Recommendation of the *World Wide Web Consortium*, https://www.w3.org/TR/prov-dm/.

Moreau, L. and Missier, P. (2013b). PROV-N: The Provenance Notation. Recommendation of the *World Wide Web Consortium*, https://www.w3.org/TR/prov-n/.

Moreau, L., Plale, B., Miles, S., Goble, C., Missier, P., Barga, R., Simmhan, Y., Futrelle, J., McGrath, R. E., Myers, J., Paulson, P., Bowers, S., Ludaescher, B., Kwasnikowska, N., Van den Bussche, J., Ellkvist, T., Freire, J., and Groth, P. (2008b). The Open

Provenance Model (v1.01). Published specification, https://eprints.soton.ac.uk/266148/.

Nesterenko, L. (2016). Building a system for stock news generation in russian. In *Proceedings of the 2nd International Workshop on Natural Language Generation and the Semantic Web (WebNLG 2016)*, pages 37–40.

Open Data Institute and Thomson Reuters (2014). Creating Value with Identifiers in an Open Data World. Technical report. https://innovation.thomsonreuters.com/en/labs/data-identifiers.html.

Packer, H. S. and Moreau, L. (2015). Sentence templating for explaining provenance. In Ludäscher, B. and Plale, B., editors, *Provenance and Annotation of Data and Processes: 5th International Provenance and Annotation Workshop, IPAW 2014, Cologne, Germany, June 9-13, 2014. Revised Selected Papers*, pages 278–280. Springer International Publishing. https://doi.org/10.1007/978-3-319-16462-5_33.

Peter, B., Julian, M. S., Georg, R., and Felix, S. (2016). Processing document collections to automatically extract linked data: semantic storytelling technologies for smart curation workflows. In *Proceedings of the 2nd International Workshop on Natural Language Generation and the Semantic Web (WebNLG 2016)*, pages 13–16.

Power, R., Scott, D., and Bouayad-Agha, N. (2003). Document Structure. *Computational Linguistics*, 29(2):211–260. https://doi.org/10.1162/089120103322145315.

Power, R., Scott, D., and Evans, R. (1998). What you see is what you meant: direct knowledge editing with natural language feedback. In *ECAI*, volume 98, pages 677–681.

Powers, J. and Stirtzinger, T. (2011). Lexicalizing an Ontology. In *2011 International Conference on Information and Knowledge Engineering*, pages 18–22, Las Vegas.

Provenance Working Group (2011). PROV Graph Layout Conventions. https://www.w3.org/2011/prov/wiki/Diagrams.

Rambow, O., Bangalore, S., and Walker, M. (2001). Natural Language Generation in Dialog Systems. In *First International Conference on Human Language Technology Research*, San Diego. Association for Computational Linguistics. https://doi.org/10.3115/1072133.1072207.

Ratnaparkhi, A. (1996). A Maximum Entroy Model for Part-Of-Speech Tagging. In *Conference on Empirical Methods in Natural Language Processing*, pages 1–10, New Brunswick, NJ. Association for Computation Linguistics.

Reiter, E. (1994). Has a Consensus NL Generation Architecture Appeared, and is it Psycholinguistically Plausible? In *The Seventh International Workshop on Natural*

Language Generation, pages 163–170, Kennebunkport, ME. Association for Computational Linguistics.

Reiter, E. (2007). An Architecture for Data-to-Text Systems. In *Eleventh European Workshop on Natural Language Generation*, pages 97–104, Schloss Dagstuhl, Germany.

Reiter, E. (2011). Task-Based Evaluation of NLG Systems: Control vs Real-World Context. In *UCNLG Eval Language Generation and Evaluation Workshop*, pages 28–32.

Reiter, E. and Dale, R. (1997). Building Applied Natural Language Generation Systems. *Natural Language Engineering*, 3(1):57–87.

Reiter, E. and Dale, R. (2000). *Building Natural Language Generation Systems*. Cambridge University Press, Cambridge.

Reiter, E., Robertson, R., and Osman, L. M. (2003). Lessons from a failure: Generating tailored smoking cessation letters. *Artificial Intelligence*, 144(1-2):41–58. `https://doi.org/10.1016/S0004-3702(02)00370-3`.

Richardson, D. P. and Moreau, L. (2016). Towards the domain agnostic generation of natural language explanations from provenance graphs for casual users. In Mattoso, M. and Glavic, B., editors, *Provenance and Annotation of Data and Processes: 6th International Provenance and Annotation Workshop, IPAW 2016, McLean, VA, USA, June 7-8, 2016, Proceedings*. Springer International Publishing. `https://doi.org/10.1007/978-3-319-40593-3_8`.

Richardson, D. P., Moreau, L., and Mott, D. (2014). Beyond the graph: Telling the story with PROV and Controlled English. In *2014 Annual Fall Meeting of the International Technology Alliance*, Cardiff, UK.

Richardson, D. P., Moreau, L., and Mott, D. (2015). What's in a name? Exploiting URIs to enrich provenance explanations in plain English. In *3rd Annual Fall Meeting of the International Technology Alliance*, College Park, MD, USA.

Saenger, P. (1997). *Space between words: The origins of silent reading*. Stanford University Press.

Sahoo, S. S., Groth, P., Hartig, O., Miles, S., Coppens, S., Gil, Y., Moreau, L., Zhao, J., and Panzer, M. (2010). Provenance Vocabulary Mappings. Design note. `https://www.w3.org/2005/Incubator/prov/wiki/Provenance_Vocabulary_Mappings`.

Sahoo, S. S. and Sheth, A. P. (2009). Provenir Ontology: Towards a Framework for eScience Provenance Management. In *Microsoft eScience Workshop*, Pittsburgh, PA, USA. Wright State University.

Sanby, L., Todd, I., and Keet, C. M. (2016). Comparing the template-based approach to gf: the case of afrikaans. In *ceedings of the 2nd International Workshop on Natural Language Generation and the Semantic Web (WebNLG 2016)*.

Searle, J. R. (1969). *Speech acts: An essay in the philosophy of language*. Cambridge university press.

Shackell, G. H. (2008). Traceability in the meat industry–the farm to plate continuum. *International Journal of Food Science and Technology*, 43:2134–2142. https://doi.org/10.1111/j.1365-2621.2008.01812.x.

Simmhan, Y., Groth, P., and Moreau, L. (2011). The Third Provenance Challenge on Using the Open Provenance Model for Interoperability. *Future generation computer systems*, 27(6):737–742. https://doi.org/10.1016/j.future.2010.11.020.

Simmhan, Y. L., Plale, B., and Gannon, D. (2005). A Survey of Data Provenance in e-Science. 34(3):31–36.

Sleimi, A. and Gardent, C. (2016). Generating paraphrases from dbpedia using deep learning. *WebNLG 2016*, page 54.

Sporny, M., Longley, D., Kellogg, G., Lanthaler, M., and Lindstrom, N. (2014). A JSON-based Serialization for Linked Data. Recommendation of the *World Wide Web Consortium*, https://www.w3.org/TR/json-ld/.

Student (1908). The probable error of a mean. *Biometrika*, 6(1):1–25. https://doi.org/10.2307/2331554.

Sun, X. (2008). *Domain Independent Generation from RDF Instance Data*. PhD thesis, University of Aberdeen.

Sun, X. and Mellish, C. (2006). Domain Independent Sentence Generation from RDF Representations for the Semantic Web. In *Combined Workshop on Language-Enabled Educational Technology and Development and Evaluation of Robust Spoken Dialogue Systems*, Riva del Garda, Italy.

Sun, X. and Mellish, C. (2007). An Experiment on "Free Generation" from Single RDF triples. In *Workshop on Syntax and Structure in Statistical Translation*. http://dl.acm.org/citation.cfm?id=1610181.

Toniolo, A., Wentao Ouywang, R., Dropps, T., Oren, N., Norman, T. J., Srivastava, M., Allen, J. A., de Mel, G., Sullivan, P., Mastin, S., and Pearson, G. (2014). Assessing the credibility of information in collaborative intelligence analysis. In *2014 Annual Fall Meeting of the International Technology Alliance*.

Trienekens, J. and Zuurbier, P. (2008). Quality and safety standards in the food industry, developments and challenges. *International Journal of Production Economics*, 113(1):107–122. https://doi.org/10.1016/j.ijpe.2007.02.050.

UK Cabinet Office (2013). Improving the transparency and accountability of government and its services.

van Deemter, K., Krahmer, E., and Theune, M. (2005). Real versus Template-Based Natural Language Generation: A False Opposition? *Computational Linguistics*, 31(1):15–23. https://doi.org/10.1162/0891201053630291.

Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., and Saenko, K. (2015). Sequence to sequence - video to text. In *The IEEE International Conference on Computer Vision (ICCV)*.

W3C OWL Working Group (2012). OWL 2 Web Ontology Language Document Overview (Second Edition). Recommendation of the *World Wide Web Consortium*, https://www.w3.org/TR/owl2-overview/.

Wilcoxon, F. (1945). Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6):80–83. https://doi.org/10.2307/3001968.