Path-Based Interactive Visual Exploration of Knowledge Graphs

Marie Destandau

université PARIS-SACLAY

Path-Based Interactive Visual Exploration of Knowledge Graphs

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 580, STIC Sciences et technologies de l'information et de la communication
Spécialité de doctorat: Informatique
Unité de recherche: Université Paris-Saclay, CNRS, Laboratoire de recherche en informatique, 91405, Orsay, France
Référent: Faculté des sciences d'Orsay

Thèse présentée et soutenue en visioconférence totale le 18/12/2020 par

Marie Destandau

Composition du jury:

Michèle SEBAG Directrice de recherche, Univ. Paris-Saclay, CNRS, Inria, LRI	Présidente
Sihem AMER-YAHIA Directrice de recherche, CNRS, Univ. Grenoble Alpes	Rapporteur & examinatrice
Roberto GARCÍA GONZÁLEZ Professeur associé, Univ. de Lleida	Rapporteur & examinateur
Hala SKAF-MOLLI Professeure associée, Univ. de Nantes	Examinatrice
Nathalie HENRY RICHE Chercheuse, Microsoft Research	Examinatrice
Emmanuel PIETRIGA Directeur de Recherche, Univ. Paris-Saclay, CNRS, Inria, LRI	Directeur de thèse
Jean-Daniel FEKETE Directeur de Recherche, Univ. Paris-Saclay, CNRS, Inria, LRI	Co-encadrant
Alain GIBOIN Chercheur émérite, Wimmics, Inria	Invité

Thèse de doctorat

NNT: 2020UPASG063

PUBLICATIONS

Journal paper

1. **Marie Destandau**, Caroline Appert and Emmanuel Pietriga. S-Paths: Set-Based Visual Exploration of Linked Data Driven by Semantic Paths. *Semantic Web Journal*. doi: 10.3233/SW-200383.

2. **Marie Destandau** and Jean-Daniel Fekete. The Missing Path: Analysing Incompleteness in Knowledge Graphs. arXiv:2005.08101. *Information Visualization*.

Conference paper

3. Emmanuel Pietriga, Hande Gözükan, Caroline Appert, **Marie Destandau**, Šejla Čebirić, François Goasdoué and Ioana Manolescu. Browsing Linked Data Catalogs with LODAtlas. *ISWC — International Semantic Web Conference 2018*.

Workshop paper

4. **Marie Destandau** and Jean-Daniel Fekete. Diagnosing Incompleteness in Wikidata with The Missing Path. *WikiWorkshop 2020, hosted by The Web Conference 2020*.

Doctoral symposium

5. **Marie Destandau**. Interactive Visualisation Techniques for the Web of Data. *The Web Conference 2020*. doi: 10.1145/3308560.3314189.

Extended Abstracts

6. Raphaëlle Lapôtre, **Marie Destandau** and Emmanuel Pietriga. Proposing rich views of linked open data sets : the S-paths prototype and the visualization of FRBRized data in data.bnf.fr. *SWIB Semantic Web in Libraries, Nov 2019, Hamburg, Germany*

7. Raphaëlle Lapôtre, **Marie Destandau** and Emmanuel Pietriga. Composant de visualisation de données liées pour la recommandation de contenus. *Atelier INRIA Culture, Nov 2018, Paris, France*

Preprint

8. **Marie Destandau**, Olivier Corby, Jean-Daniel Fekete and Alain Giboin. Path Outlines: Browsing Path-Based Summaries of Linked Knowledge Graphs. arXiv:2002.09949.

ABSTRACT

Keywords: Knowledge Graphs, Visualisation, RDF, Interactive Exploration, Semantic Web, Linked Data

Knowledge Graphs facilitate the pooling and sharing of information from different domains. They rely on a flexible encoding format, the Resource Description Framework (RDF), which enables describing and connecting heterogeneous data sources despite their differences. RDF data consist of simple statements, named triples, that can be chained to form higher-level statements across datasets, following information needs. Producing interactive visual interfaces to explore Knowledge Graphs is a complex problem, mostly unresolved, for two main reasons: 1) meaningful information to describe a collection can be several triples away from the entities; and 2) entities in a collection are not necessarily described with similar properties. In this thesis, I introduce the concept of *semantic paths* to encode aggregate information relative to a chain of triples; they consider missing information as part of the description, thus providing a common space to characterise heterogeneous resources of various depths. I first use this concept to automate the production of meaningful overviews for any set of entities. I design and implement S-Paths, an open-source browser to let users navigate through RDF collections, starting with an overview of a whole collection, and offering new overviews of the subsets as they progress through the collection, refining their selection. I report a qualitative evaluation showing that they can make sense of such overviews and remember the important dimensions of a dataset in the main lines. Then, I reuse the concept—renamed *path outlines* to better convey the idea of a summary—to produce path-based summaries of RDF datasets. I interview 11 data producers to confirm their interest. I design and implement Path Outlines, an open-source tool based on coordinated views with two novel visualisations, the *broken (out)lines* and the *path browser*, to support RDF data producers in browsing the statements that can be produced from their dataset. I compare Path Outlines with a SPARQL query editor, the current baseline technique to access path-based information, in a controlled experiment with 36 participants. I show that Path Outlines is 3 times faster, leads to better task completion, fewer errors, that participants prefer it, and find tasks easier and more comfortable with it. Finally, I apply this concept

to support data producers in analysing incompleteness in their data. I design and implement *The Missing Path*, an open-source visualisation tool to help users analyse incompleteness for groups of entities. It computes clusters with similar incomplete profiles on a map and lets users inspect and contextualize their statistical summaries. I conduct an iterative design process and evaluation with Wikidata contributors. Participants gain insights and find strategies to identify coherent subsets to be fixed, using the coordinated views in various exploratory ways, starting from the map or the summaries. With those 3 applications, I show that *path outlines*, overcome some of the complexity at the heart of RDF, and not only support interactive visual interfaces for Knowledge Graphs but also help better their quality. I can foresee other applications of the concept, such as tools to design ontologies or exploratory analysis tools.

SYNTHÈSE

Mots-clés: Graphes de connaissance, Visualisation, RDF, Exploration interactive, Web sémantique, Données liées.

Les Graphes de Connaissances représentent, connectent, et rendent interprétables par des algorithmes des connaissances issues de différents domaines. Ils reposent sur un format d'encodage flexible, RDF (Resource Description Framework), qui supporte la description et l'interconnexion de sources de données hétérogènes malgré leurs différences. RDF est basé sur des énoncés simples, nommés triplés, que l'on peut chaîner pour former des énoncés de plus haut niveau, en passant d'un jeu de données à un autre, en fonction des besoins d'information. Produire des interfaces visuelles interactives génériques pour explorer des collections dans des Graphes de Connaissances est un problème complexe, en grande partie non résolu, pour deux raisons principales: 1) l'information pertinente pour décrire une collection peut se trouver au bout d'une chaîne de plusieurs triplés; et 1) les entités d'une collection ne sont pas toujours décrites avec des propriétés identiques. Dans cette thèse, je propose le concept de chemins sémantiques pour décrire les énoncés de haut niveau: il encode des informations agrégées relatives à une chaîne de triplés, et considère l'information manquante comme partie intégrante de la description, produisant ainsi un espace commun pour caractériser des ressources hétérogènes de profondeur variée. Dans un premier temps, j'utilise ce concept pour générer des vues d'ensemble de façon automatique pour tout ensemble d'entités. Je designe et implémente S-Paths, un navigateur open source qui permet d'explorer des collections en partant d'une vue de toute la collection, présentant de nouvelles vues synthétiques à mesure que les utilisateurs affinent leur sélection. Je relate une évaluation qualitative montrant que les participants arrivent à interpréter ces vues d'ensemble et mémorisent les dimensions importantes du jeu de données dans les grandes lignes. Dans un deuxième temps, j'utilise ce concept—renommé profils de chemins afin de mieux traduire l'idée de résumé—pour produire des résumés basés sur les chemins. J'interviewe 11 producteurs de données pour confirmer leur intérêt. Je designe et implémente Path Outlines, un outil open source basé sur des vues coordonnées avec 2 nouvelles visualisations, les lignes brisées et le navigateur de chemins, pour permettre aux producteurs

de données RDF de parcourir les énoncés qui peuvent être produits par leurs jeux de données. Je le compare à un éditeur de requêtes SPARQL, la technique de référence pour accéder à des informations basées sur les chemins, dans une expérience contrôlée avec 36 participants. Je montre qu'il est 3 fois plus rapide, mène à un meilleur accomplissement des tâches et moins d'erreurs, que les participants le préfèrent et trouvent les tâches plus faciles et plus comfortables quand ils l'utilisent. Dans un troisième temps, j'applique ce concept à l'analyse de l'incomplétude d'un jeu de données. Je designe et implémente The Missing Path, un outil de visualisation open source basé sur des vues coordonnées pour permettre aux producteurs de données d'analyser l'incomplétude de sous-ensembles. L'outil calcule des groupes avec des profils similaires et les dispose sur une carte. Les utilisateurs peuvent sélectionner un sous-ensemble, inspecter sa distribution, et le situer dans le contexte du jeu de données complet. Je rapporte un processus de design itératif et une évaluation qualitative avec 9 contributeurs Wikidata. Les participants acquièrent des connaissances et trouvent des stratégies pour identifier des sous-ensembles cohérents à réparer, en utilisant les vues coordonnées de diverses manières exploratoires, à partir de la carte ou des résumés. À travers ces 3 applications je montre que les profils de chemins, en abstrayant une partie de la complexité de RDF, permettent non seulement de supporter des interfaces visuelles interactives, mais aussi d'améliorer la qualité des Graphes de Connaissances. J'envisage d'autres applications pour le concept, telles que des outils pour le design d'ontologies ou l'analyse exploratoire.

REMERCIEMENTS

Cela pouvait sembler insensé de se lancer dans une thèse à 37 ans, sans diplôme de master. Merci à tous ceux qui ont considéré ce projet sérieusement, en dépit du bon sens, et m'ont aidée à l'amorcer. En particulier, merci à Pierre Marie de m'avoir aidée à trouver comment surmonter tous les obstacles. Merci à Rodolphe Bailly de m'avoir donné l'opportunité de rejoindre le projet Doremus. Merci à tous les membres du projet Doremus, et à tous les collègues de la Philharmonie. Merci à Raphaël Troncy et Konstantin Todorov de m'avoir enseigné les bases du Web Sémantique, donné l'opportunité de participer à leurs papiers, et écrit des lettres de recommandation généreuses. Merci à Marie Després-Lonnet d'avoir supervisé mon mémoire de master, me permettant d'obtenir la mention indispensable pour un financement. Merci à Bertrand Sérieyx de m'avoir présentée à Camille Picard qui m'a orientée vers l'INRIA. Merci à Alain Vagner de m'avoir appris l'existence de Google Scholar, m'ouvrant les portes d'un nouveau monde. Merci à Gérard Denis pour le TOEFL. Merci à Hugues Moreno, Viviane Roth et Marie de Ramefort. Et merci à tous ceux qui m'ont encouragée, d'une façon ou d'une autre.

Les trois années qui ont suivi se sont révélées encore plus insensées. Merci à tous ceux qui m'ont aidée à tenir bon et à aller jusqu'au bout. Merci à Jean-Daniel Fekete de m'avoir recueillie dans son équipe de choc, de m'avoir appris à écrire un papier et une revue, de m'avoir encouragée à avoir l'air plus sûre de mon travail, et d'avoir finalement dirigé cette thèse. Merci à tous les membres d'Aviz: Natkamon Tovanich, Pierre Dragicevic, Petra Isenberg, Catherine Plaisant, Paola Valdivia, Sarkis Halladjian, Tanja Blascheck, Frédéric Vernier, Mickaël Sereno, Xiyao Wang, Tanja Blascheck, Lonni Besancon, Katia Evrat, Gaëlle Richer, Yuheng Feng, Jiayi Hong, Alexis Pister, Alaul Islam, Tobias Isenberg et Steve Haroz. Merci à ILDA, et en particulier à Anna Gogolou, Adhitya Kamakshidasan, Maria Lobo, Arnaud Prouzeau, Bruno Fruchard et Dylan Lebout. Merci à Ex-Situ, et particulièrement à Wendy Mackay et Michel Beaudoin-Lafon pour leurs formidables cours et séminaires, et à Jean-Philippe Rivière et Yi Zhang. Merci à Raphaelle Lapôtre et Aude Le Moullec-Rieu, de la Bibliothèque nationale de France, pour leurs appréciations éclairées et leur soutien. Merci à Juliette pour les dessins. Merci à Wimmics, en particulier à Alain Giboin et Olivier Corby pour la collaboration sur *Path*

Outlines, et à Marco Winckler, Franck Michel, Michel Buffa et Fabien Gandon pour leur feedback. Merci à Tiziana Catarci. Merci à Staff pour le support technique toujours parfaitement aimable et diablement efficace: Laurent Darré, Vincent Néri, Denis Humbert et Anthony Pensel. Merci à Christian Poli pour Gunicorn. Merci à Edoardo Cecchin pour les vidéos, à Valentina Pérez Llosa et Markus Detmer pour la voix. Merci à Silvio Cardoso pour la super expérience d'enseignement partagé. Merci à Nicolas Ferey et Julien Nelson. Merci à Ben Steichen pour ses conseils précieux au Consortium Doctoral de TheWebConf. Merci à Thomas Minier, Hala Skaf-Molli, Pascal Molli, Ahmed El Amine Djebri, Lionel Medini, Sébastien Desbenoit et Margaret Warren pour la super semaine partagée à San Francisco. Merci à ma mentore en or Julie Menetrey de m'avoir aidée, soutenue, et transmis des clés salutaires. Merci à l'Association Femmes et Sciences et à l'équipe du mentorat. Merci à Valérie Berthou, Viviane Gamboa, Nicolas Anciaux, Anne Brun. Merci à Stefan Münnich. Merci à Odile Duplessis pour les super formations. Merci à Gert Rietveld pour ses conseils. Merci à Neil et Pam pour la relecture.

Merci à tous les participants des ateliers et expériences. Je ne les nommerai pas, car leur participation était anonyme, mais je tiens a leur dire que je me souviens précisément de chacun d'entre eux, que j'ai été vraiment impressionnée par leurs compétences et qualités et que leurs retours m'ont beaucoup fait avancer.

Merci à Logilab—chez qui j'ai passé quelques mois à attendre une réponse venue trop tard pour faire cette thèse en CIFRE—, et notamment à Tanguy Le Carrour pour les cours de Python, JS et TDD, et à Adrien Di Mascio, Marla Da Silva, David Douard et Nicolas Chauvat.

Merci à ma famille, avec une mention spéciale pour mes cousines Céline, Clorinde et Clémence, leurs moitiés, et Sarah et Nico. À la mémoire de Monique et de Josy, dont le regard bienveillant et la générosité m'accompagnent au quotidien, malgré leur absence.

Merci à mes amis, et particulièrement à Marie, Christophe, Stéphane V., Ellen, Ghislaine, Elisabeth, Germain, Pierre, Charlotte, Julien, Stéphane A., Hélène, Guy, Marie-Anne, Domi, Laurence et Blaise, Jérôme, Hugues, Liselotte et Charles.

Merci infiniment à Flo, mon merveilleux mari.

CREDITS

***S-Paths* prototype**

Icons: Data Visualization Icon Set and The Noun Project

***The Missing Path* prototype**

Pictograms by The Noun Project.

Manuscript

Drawings in Fig. 2, Fig. 3, Fig. 4, Fig. 30, Fig. 32, Fig. 35,Fig. 41 and Fig. 56 by Juliette Taka.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

We cannot solve our problems
with the same level of thinking
that created them.

---

attributed to Albert Einstein

1

INTRODUCTION

Knowledge Graphs (KG) [53] are everywhere. The technology is underlying in our everyday life, powering search engines [125], recommender systems [129] and connected objects [67]. The power of KG comes from their simple and flexible encoding format, the Resource Description Framework (RDF), that allows to describe and connect heterogeneous data sources. Merging data from different domains increases their value, which is "directly proportional to the interlinkedness of the data" [88]. Companies use them to carry out global corporate knowledge management strategies, with applications ranging from risk management to process and factory monitoring [57]. Institutions [111] and communities [36] rely on them to publish and share their data in an interoperable format. Digital Humanities researchers use them to support exploration and analysis of large corpus connecting various sources and are starting to investigate machine learning assisted applications such as knowledge discovery and computational creativity [60]. In the context of smart cities, they support the management of various indicators acquired from a multitude of sources including instruments, sensors, humans, and computer models [105]. When published on the web, RDF data can be queried jointly through federated queries, forming a network of interlinked sources, known as the Web of Data or the Semantic Web.

Although the technology is developing rapidly, it faces a difficulty that threatens the quality of data: the lack of generic interfaces to browse a dataset. If RDF data are designed to be processed by algorithms, humans still need to interact with them at some point, at least to control their production, and to design and develop the algorithms to consume them. Developing a specific application for each dataset has a cost, and takes time. As a result, most of the time, accessing RDF data implies reading raw data files with hundreds of thousands of lines or querying them with SPARQL, the query language for RDF data. This requires technical skills, time and concentration, and hinders the detection of errors and irregularities.

As I worked for Philharmonie de Paris[1] on a research project aiming to transform into RDF and interlink the musical catalogues of 3 institutions [3],

1 philharmoniedeparis.fr

from 2015 to 2016, I was struck that the data producers could not see their data. From the moment the Modeling Working Group delivered the mapping rules for the transformation of original data sources, it took more than a year to develop a specific interface to visualise them. In the meantime, they had to rely on their knowledge of original data, of mapping rules and of target ontologies to imagine what the result would be. Though they were librarians, with an impressive ability to think abstractly, and a precise knowledge of the original data, this was constantly raising difficulties and misunderstandings. They spent a significant amount of time trying to represent samples of data in hand made node-link diagrams and spreadsheets to be able to work together. At this time, I also met other RDF data managers and developers in meetups and professional conferences, and I realised that many of them were facing similar problems. I tried to develop a browser for SKOS thesauri. In a naïve approach, I used a tree-like node-link diagram layout, matching the structure of data, and had to acknowledge that it did not scale, neither graphically nor from a performance point of view. With 10 years of experience developing web applications, I could not find a satisfying solution. And I was only addressing the specific case of SKOS thesauri, for which the structure is more or less known in advance. This experience provided the motivation for this thesis.

Starting this research, I discovered that there is actually an impressive number of tools, but that they have limitations preventing their effective use. RDF data are graph data, based on simple statements, named triples, that can be chained to form higher-level statements. For instance, ‘Marie Curie is affiliated to Sorbonne University’ and ‘Sorbonne University is located in Paris’ are two triples, that can be chained to produce the statement ‘Marie Curie is affiliated to Sorbonne University in Paris’. Producing interactive visual interfaces to explore Knowledge Graphs is a complex problem, mostly unresolved, for two main reasons: 1) meaningful information to describe a collection (e.g. Nobel laureates) can be several triples away from the entities (the laureates); and 2) entities in a collection are not necessarily described with similar properties (e.g. *affiliated to* or *located in*).

On the one hand, RDF browsers are focused on single entities, displaying one page per entity, and letting users hop in the graph one step at a time. Gaining information about a collection requires to mentally do a synthesis of little pieces of data, gathered one after one another, which entails a substancial cognitive load. On the other hand, visualisation systems produce overviews, but involving only a few direct properties, and with little support for navigation. RDF data seem to defeat the Information Seeking Mantra,

'Overview first, zoom and filter, then details-on-demand' [113], that has shaped generations of interfaces to browse collections of data. In the related flourishing research on graph visualisation, state of the art interfaces for graphs do not scale to even very small RDF dataset, producing cluttered interfaces, both unreadable and unusable [109]. Parallel coordinates and matrixes accept with a limited number of attributes and values. Techniques to make node-link diagrams scale, such as edge bundling, work only with very simple datasets encoding a single type of relation. The representation of a RDF graph, with many items and unpredictable attributes of various types and at various depths, is still a puzzle.

Therefore, my main research question is: **how to design generic interactive interfaces to visualise collections of resources in RDF datasets, and browse them from overview to detail?**

1.1 THESIS STATEMENT

I argue that a level of granularity that matches human understanding is needed to explore collections in Knowledge Graphs: a concept to describe higher-level statements in a systematic way, that will relieve the human brain from the combinatory work needed to reconstitute chains of triples. I present the concept of *semantic path*—also named *path outline*, to encode aggregate information relative to chains of triples. By considering missing information as part of the description, it provides a common space to describe heterogeneous resources of various depths. I demonstrate 3 applications of this concept: a browser producing overviews of collections and their subsets to support navigation through a dataset, a tool to browse the possible statements in a dataset, and a tool to analyse incompleteness in a dataset.

1.2 THESIS OVERVIEW

In *Chapter 2* BACKGROUND AND RELATED WORK, I first introduce RDF data, their structure and the difficulties to visualise them. Then I present 3 types of users who need to interact with raw RDF datasets. I review existing tools to browse and visualise RDF data.

In *Chapter 3* S-PATHS, I first address the specific question **how to produce meaningful overviews of a collection in an RDF dataset, as well as of its subsets at different scales?** I introduce the concept of *semantic paths*, to provide a summary of all properties or chain of properties describing a set of entities, and use it to power *S-Paths*, a semi-automatic browser providing synthetic views of a collection. I present its design and architecture. I report

a real use case with the data of the french national library, and a qualitative user study with 6 users.

Motivated by the reaction of the product manager of data.bnf.fr[2] to *S-Paths*, *Chapter 4* PATH OUTLINES addresses another specific question: **how to visualise the statements produced by an RDF dataset, and browse them by meaningful chunks?** I present *Path Outlines*, an open-source tool to support RDF data producers in browsing the statements that can be produced from their dataset. It displays *semantic paths*—that I rename *path outlines*, to better convey the idea of a summary—through coordinate views with two novel visualisations, the *broken (out)lines* and the *path browser*. I report a controlled study comparing *Path Outlines* with the current baseline technique (Virtuoso SPARQL query editor) in an experiment with 36 participants.

Finally, addressing a limitation of *Path Outlines*, *Chapter 5* THE MISSING PATH investigates the specific question: **how to support visual analysis of the incompleteness of a collection in an RDF dataset, and of its subsets at different scales?** I present *The Missing Path*, an open-source visualisation tool to support users in analysis incompleteness for groups of entities. It uses *path outlines* to compute clusters with similar incomplete profiles and lay them down on a map, letting users select any subset, inspect its distribution, and compare it to the full dataset. I conduct an iterative design process and evaluation with 9 Wikidata contributors.

Finally, in *Chapter 6* CONCLUSION AND DISCUSSION, I summarise my contributions and discuss other applications of the concept that I can foresee.

1.3 CONTRIBUTIONS

1.3.1 *Theoretical contributions*

I introduce **the concept of *semantic path* / *path outline***: for a given collection, it encodes aggregate information relative to a chain of triples, considering missing information as part of the description, thus providing a common space to characterize heterogeneous resources of various depths. First designed as an abstraction to program *S-Paths* browser, its graphical representation proved useful to support tasks which necessitate to understand the structure of the content, with *Path Outlines* and *The Missing Path*.

1.3.2 *Technological contributions*

I design and implement:

2 the Linked Data service of the French National Library

- ***S-Paths*, a browser providing synthetic views of a collection and any of its subset**: s-paths.lri.fr. A matching algorithm compares the *path outlines* of the collection with the requirements of a set of views optimised to provide synthetic visualisations at various scales, and provides a default view to users. The code is open-source: gitlab.inria.fr/mdestand/s-paths;
- **an API to analyse *path outlines***. It is open-source. It is included in *S-Paths* package but can be run separately[3];
- ***Path Outlines*, a tool to support RDF data producers in browsing the statements that can be produced from their dataset**: spf.lri.fr. It is based on coordinate views with two new visualisations: the *broken (out)lines* and the *path browser*, to display the *path outlines*. The code is open-source: gitlab.inria.fr/mdestand/spf;
- ***The Missing Path*, an open-source visualisation tool to support users in analysis incompleteness for groups of entities**: missing-path.lri.fr. It computes clusters with similar missing *path outlines* and lays them down on a map, letting users select any subset, see the level of completeness of all its *path outlines*, relate it to the distribution of their values, and situate it in the context of the full dataset. The code is open-source: gitlab.inria.fr/mdestand/the-missing-path.
- **an API to extract the content of a collection in a matrix driven by *path outlines***, to support responsive processing of the content. It is included in *The Missing Path* package.

1.3.3 *Empirical contributions*

I observe that:

- **users make sense of path-based overviews, remember the important dimensions of a dataset in the main lines, and are able to imagine new applications for the dataset and the tool** after 20 minutes of navigation with *S-Paths*;
- **data producers access path-based information with *Path Outlines* 3 times faster than with the baseline; it leads to better task completion, fewer errors, they prefer it, and find tasks easier and more comfortable with it**. The controlled experiment was conducted against SPARQL Virtuoso Query editor as a baseline, with 36 participants. Sup-

3 another version of the API, supporting *path outlines* running across datasets, was developed by Olivier Corby as an extension of Corese, and is used for *Path Outlines*.

plemental material is made available: ieee-dataport.org/documents/path-outlines.

- in an iterative design process and evaluation with 9 Wikidata contributors, **participants gain new insights on incompleteness in their data with *The Missing Path*, using various exploratory strategies** supported by the coordination between the map with clusters of entities, and the statistical summaries of their path.

2

BACKGROUND AND RELATED WORK

In this section, I first introduce RDF data. Then I discuss the need for interfaces to browse RDF data for three types of users: *data producers*, *data reusers* and *lay users*. I review existing approaches and tools to visualise and browse RDF data regarding their ability to browse from overview to detail. I discuss several approaches used in network visualisation, and the limitations to apply them to RDF.

2.1 INTRODUCTION TO RDF

The interoperability of Knowledge Graphs lies in the representation of information according to a common framework, the Resource Description Framework (RDF) [22]. RDF data are collections of statements named triples. Triples are composed of a *subject*, a *predicate* and an *object*, as shown in Fig. 1. Subjects can be *Uniform Resource Identifiers* (URIs) or blank nodes. Predicates are always URIs. Objects can be URIs (ℓ. 4–10), literals (e.g., strings, numbers, dates, ℓ. 1–3) or blank nodes. The same URI can be the subject and object of several triples (ℓ. 6, 7, and 8 or ℓ. 7, 9 and 10). The triples form a network. Predicates and classes of resources are defined in data models called *ontologies*. For instance, the predicates of the 3 first triples and the object of the 4th belong to the FOAF [18] (friend of a friend) ontology, dedicated to the description of people and their relationships. In principle, URIs should be dereferenceable: querying them on the web should return their RDF description. Literals can have a datatype, and string literals can be associated with a language (Fig. 1-a, grey colour). URIs can be prefixed for better readability, as in Fig. 1-c: the beginning, common to several URIs, is given a prefix (a short name), e.g. `foaf:` instead of `http://xmlns.com/foaf/0.1/`. Formally, a RDF graph is a set of triples $t = (s, p, o)$, with $s \in U \cup B$, $p \in U$ and $o \in U \cup L \cup B$. U is the set of URIs, L the set of literals, and B the set of blank nodes in the graph.

RDF data are interlinked: a dataset can reference an entity produced in another one (red colour). When this happens, a chain of statements can jump from one dataset to another: the triples in Nobel Dataset *la Sorbonne is in Paris, Paris entity in Nobel is equivalent to Paris entity in DBpedia* can be

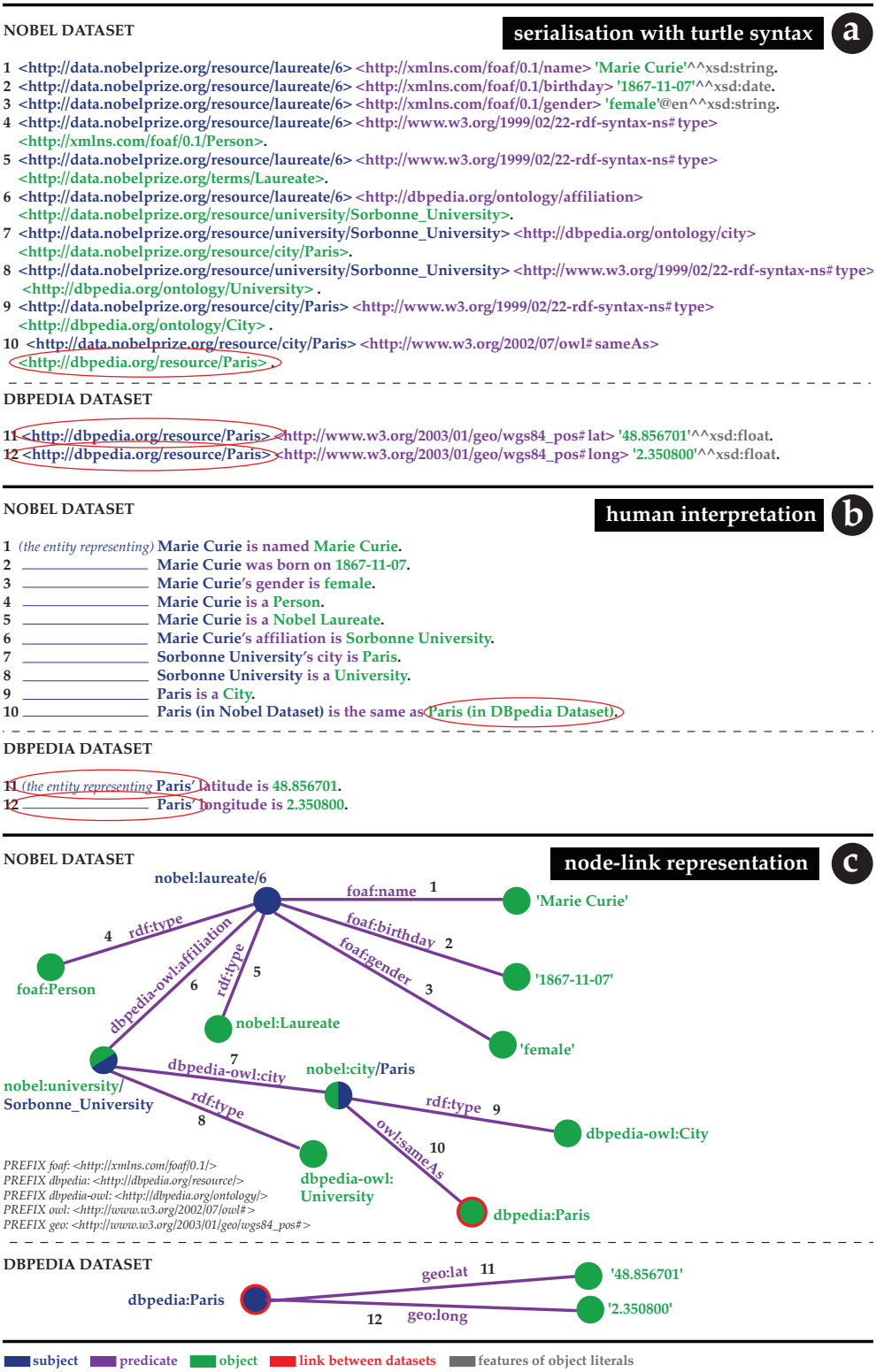


Figure 1: Samples extracted from Nobel (10 triples) and DBpedia datasets (2 triples) comparing 3 representations: a) a serialisation with Turtle syntax, which is meant for machines, but verbose and difficult to decipher for humans; b) an interpretation of each triple as a sentence understandable by humans to help understand the chaining mechanism; c) a visualisation as a node-link diagram showing how triples are connected and can be chained; it is easy to follow the paths but difficult to read their labels and make sense of them. The samples are interconnected; it is possible to combine them. Full datasets contain respectively 87,422 and 185,404,534 triples (on 2019-09-07).

completed by those in DBpedia: *Paris' latitude is 48.856701, Paris' longitude is 2.350800*. They can be queried jointly, through *federated* queries. The information is separated into atomic pieces that can be retrieved and combined following the information needs. For instance, a question like "When was Marie Curie born?", could be answered with triple 2. "What was her affiliation?" could be answered by chaining triples 6, 7 and 10. Placing Marie Curie on a map displaying laureates by affiliation could be achieved by chaining triples 6, 7 and 10 and 11 to get the latitude, and 6, 7 and 10 and 12 to get the longitude. A chain of statements is commonly called a *path* in the graph. Fig. 1-c shows a sample of 10 statements, 6 at the first level, 2 at the second and 2 at the third. In the real dataset, considering all the triples describing Marie Curie by chaining up to 3 triples, there are 672 triples, 23 at the first level, 99 at the second and 550 at the third. Even trying to represent this information mentally is difficult. The cognitive effort needed to split a piece of meaningful information (e.g., a laureate and her biographical information) into triples, and to imagine all possible combinations, is tremendous. Doing so on sets of entities (e.g., all laureates or all prizes) is even more difficult. Node-link diagrams (Fig. 1-c) are often used to explain RDF data, as they accurately render their structure.

The special predicate `rdf:type` (ℓ. 4, 7 and 8) indicates that an entity belongs to a *class of resources*. An entity can belong to several classes; for instance, the entity representing Marie Curie belongs to the class `foaf:Person` and to the class `nobel:Laureate`. A common designation for entities belonging to the same class is a *set of entities*. In this thesis, as I aim at making RDF accessible to humans, I use the word *collection*, which conveys more semantics and implies that the set has a meaning for users. This facilitates reading and avoids abstract sentences. There are other ways to define collections, such as a pattern in a query, or specific ontologies like PROV-O. RDF Semantics recommendation uses the term collection for 'list structures', that is a set of untyped entities. The word collection, when used in this thesis, refers to a set of entities belonging to the same class of resources. Supporting other types of collection might necessitate an adaptation of the query templates.

2.2 RDF FOR HUMANS

Though RDF data are originally meant to be processed automatically, there are many situations where humans need to interact with them, be it only to design and control the algorithms to produce and consume them. In the first

place, I am particularly interested in who, as of today, are constrained to interact with raw datasets to find out what is inside.

In the early design phase of *S-Paths*, we conducted two workshops in order to inform our design choices—the reports are available in Appendix B. The first workshop involved 9 Linked Data experts, and the second 7 lay users who were interested in exploring RDF data. We asked them to sketch scenarios of what they would like to be able to do when exploring RDF data, illustrating how an ideal tool would support them. From these workshops, combined with informal observations during the French National Library's hackathon and other community events in Paris, I derived three persona representing my target users: a *data publisher*, a *data reuser*, and a *lay user*.

Figure 2: RDF data producer: Bob is trying to figure out what his own dataset contains.

2.2.1 *Data producers*

Before algorithms can consume RDF data, the data have to be produced. RDF data producers can be *data manager* in an IT department, with a strong technical background, allowing them to query and transform the data themselves. However, as the technology spreads, it becomes more usual to meet an RDF *data curator* or *product manager* with an editorial profile and expertise about the domain, but less developed technical skills. Librarians, who are

very active in the Semantic Web, organising professional conferences that are very popular and well attended, are a good example of the latter. Researchers in Digital Humanities, such as historians and musicologists, have also started to convert their data to RDF to beneficiate from the possibilities of analysis offered by the format. The level of their technical skills is very uneven.

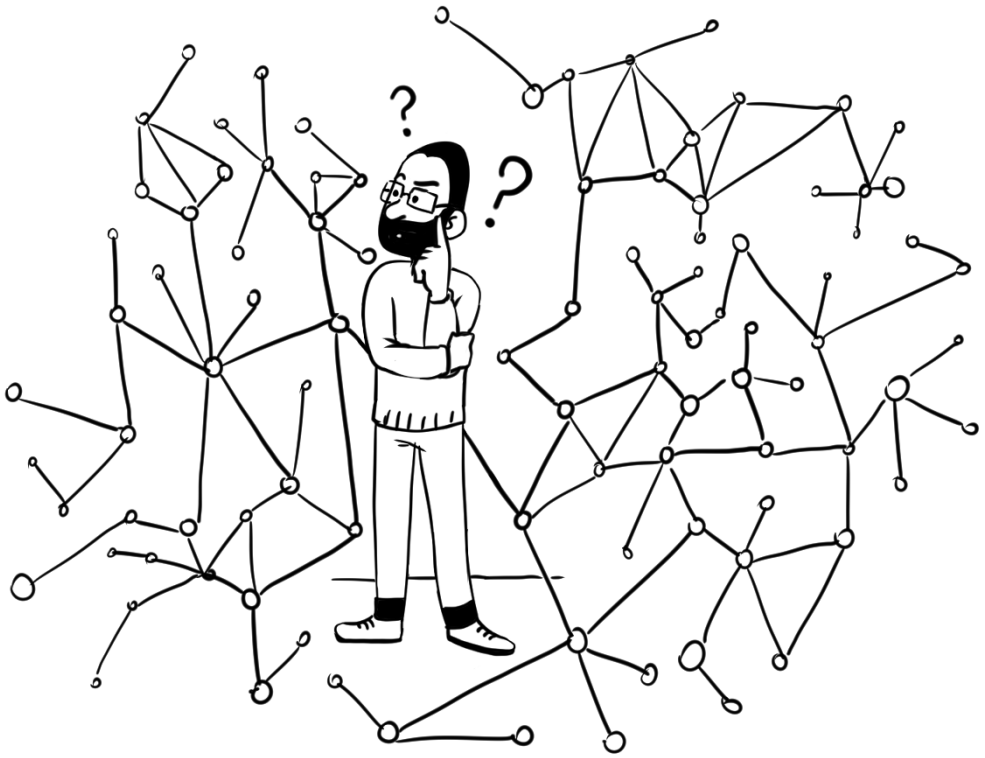


Figure 3: RDF data reuser: Tim is trying to understand how the information he wants to access is expressed.

Since there are few tools to manage RDF content natively, RDF data are most of the time created from the transformation of data sources originally stored in another format, and often aggregate information from various sources. Data producers need to control the result of the transformation and merge, gaining an overview and inspecting the details. In particular, they need to see the possible statements produced by the graph, to check that the data are efficiently described and make sense, and to manage the heterogeneity.

To promote their Knowledge Graphs and be able to describe the content, they also need an overview of the trends and interesting fact in their data.

Persona. *Bob is in charge of the Linked Data published by the French National Library. He is a librarian, graduated from Ecole des Chartes, where he received basic education and training about Linked Data. After an internship in a city library, he was recruited by the BnF 5 years ago. He then perfected*

her skills at the library. He regularly exchanges skills with his colleagues, engineers and librarians. They all participate in events organized by and for librarians around the Bibliographic Transition, which happens in close relation to semantic web technology, and also to general international and national librarian events. Being in charge of service providers for Linked Data, he can rely on them for specific questions. The transfer of knowledge has happened though regular communication, as well as specific training actions. As a librarian, he has a weekly slot when he seats in the library and helps users find information, which means he is in direct contact with reusers.

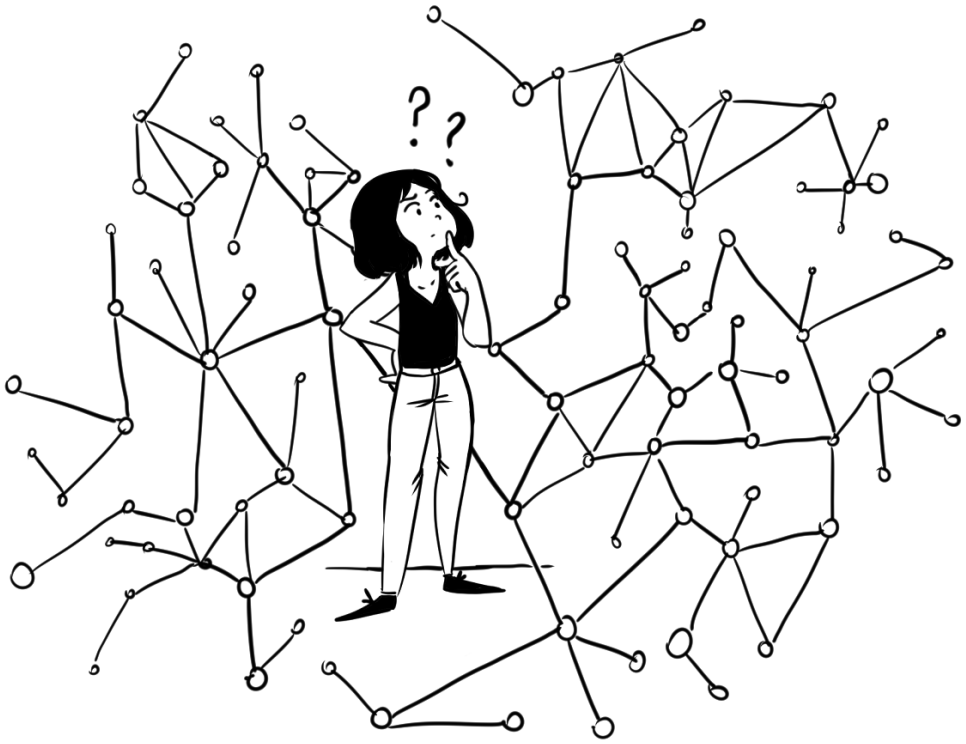


Figure 4: RDF lay user: Alice is trying to find out if the dataset published by the government contains information of interest.

2.2.2 *Data reusers*

Before algorithms can consume RDF data, the algorithms must be designed and programmed. Of course, the idea behind RDF is that, as the data embed their model and semantics, algorithms can process them with no prior knowledge of their structure. This might be valid for generic browsing and visualisation tools such as those we will review in Sect. 2.3. But there is no such thing as a generic algorithm that could power specific application as diverse as corporate knowledge management systems, social networks, IOT programs or recommender systems.

To design such an application, its features, interface and interaction flow, applications designers need an overview of the information which is available in the data, and how it is expressed.

To develop such an application, application developers need to profile the data: see how to access a piece of information, and the outlines of its possible values.

Persona. *Tim is R&D manager for a small company publishing software for libraries. They use Linked Data to provide more services in their software. It is a growing concern for libraries, and as such a crucial feature for the company. Today, it is still a competitive argument; tomorrow, it might be decisive for the survival of the company. Since it's still an immature technology, his technological watch requires particularly active involvement. Besides reading scientific papers, following twitter feeds, and attending professional events around the Bibliographic Transition, he also regularly participates in hackathons and similar social events. This enables him to develop his skills, as well as his network of reliable resources and potential customers. He is familiar with the SPARQL query language, which he can use to explore a Linked Dataset rapidly.*

2.2.3 *Lay users*

Once RDF data have been produced, they are—not always, but often—shared as Open Data. Non-expert users, interested in knowing about the content of a dataset, would also need a generic application to browse from overview to detail. This includes any citizen who would like to consult a dataset published by a government or an institution.

Persona. *Alice is a student in political sciences. She has just completed her first year for a Master Degree at Sciences Po Paris and is starting the second year. She would like to write a thesis about the Management of an international Fablab Community. She is curious about cultural heritage, aware of the stakes of open data, and always prompt to explore new data published by institutions. She uses such data for her studies.*

There are actually many more user types to consider in the use of RDF data. Each specific application comes with its own *end users*, with specific needs and usage. There are also *ontologists*, who create and manage the data models that will shape RDF datasets. Further applications of *paths outlines* might apply to those users, and I will mention them in Chapter 6. However, in this thesis, I focus on users who need generic tools to browse raw RDF data.

2.3 VISUALISING RDF DATA: OVERVIEW OR DETAIL

I will discuss how the different types of tools handle overviews and details, with a focus on generic tools, able to handle any RDF dataset without specific development. A comprehensive list of browsers and visualisation systems can be found in this recent book [97].

2.3.1 *RDF browsers*

The structure of RDF fosters navigation systems exploring one resource at a time[65, 99, 117, 124]. They display one page per entity, fetching its direct properties and allowing users to hop in the graph one step at a time. Those tools have the advantage to be fully generic, model agnostic, and to enable navigation across datasets. However, *follow-your-nose* navigation puts much cognitive burden on users: reaching the property of interest requires several clicks, hopping from page to page and losing the context of the original resource. Keeping the chain of previous pages in mind requires significant memorisation and integration effort. The associated mental effort increases as users browse back and forth. Haystacks has a template to display collections[99], and Noadster builds custom navigation tree for search results for users to select single resources [103]. Both give an overview of a collection but do not allow to filter it.

Faceted browsers use the properties describing a collection and their distribution to let users filter it [41, 58, 108, 137], the set of filters serving as an overview of the collection [37]. However, except with very small datasets, they present a major usability issue. RDF data are heterogeneous, the number of properties in a collection grows with the number of items, while some properties might concern very few items only. This results in cluttered and inefficient interfaces [71]. The facets might take so much space that they are no more side-navigation but represent the main part of the interface, and the collection of items becomes a component among others [7, 19, 48]. Considering that relevant properties can be several triples away from entities in the collection, some facets include paths of properties [24, 86]. While this gives access to relevant information, it also increases the number of options. This overload problem can be alleviated by enabling users to select facets on the fly [52], and help them determine their relevance through by statistical or reasoned algorithms [86]. More details about faceted browsers can be found in this survey [122]. In all cases, there is a trade-off between giving a full overview and having readable and usable navigation.

Most of the time, even after using facets to filter, the number of items to display is still too high to present them all; they are presented as paginated lists [41], or very long scrolling pages on which only the first few items are visible [9, 40, 64, 98], indicating the count of total pages and / or entities in the collection. Discovery Hub uses a sampling technique to present a representative panel of answers as an overview [72]. Facete uses a map as a continuum to display all entities, and takes advantage of paths of properties to find geographical information related to any set of entities [116].

2.3.2 *RDF visualisation systems*

Visualisation systems present charts or aggregated view as the main view, enabling to display all items along with one or a few selected properties. Those systems manage to remain generic by giving configuration work to users. SemLens lets users select any two properties to explore correlations in a unique scatterplot visualisation [50]. Tabulator requires users to write a query in SPARQL, and then automatically selects map-based or timeline-based visualisations [15]. Payola presents a graphical query builder for users to retrieve data, before presenting them a list compatible visualisations. VisWizard [104] needs users to select a visualisation first, and then properties to display in it. LinkDaVis asks users to select properties or paths of properties first, and them suggests a ranked list of visualisation configurations [119]. LinkDaVis considers not only direct properties but also chains of properties to populate the views, offering more possibilities for advanced and meaningful visualisations, but it also increases the number of choices presented to the user. In all cases, the configuration load for the user is heavy, and requires to make decisions of information before having visualised them. With the exception of Ferasat [63], those systems do not support filtering collections and progressing through datasets. Ferasat features only direct properties.

In an original approach, RelFinder explores and visualises in a node-link diagram all possible paths between two entities [49], giving at the same time overview and detail at this very specific scope.

Tools using a node-link diagram as the main interface attempt to present the full graph to let users browse it [95]. However, node-link diagrams can handle only very small datasets (Fig. 1 shows that a node-link with 11 triples already requires efforts to read). This results in ‘big fat graph’s that are not usable [109]. Another approach is to draw the graph incrementally: this works well to explore a specific portion of the graph in details, but broadening the focus requires many clicks, and the visualisation quickly becomes unread-

able [21]. Relieving the visualisation from the need to read labels, by representing only one type of entities and relations, is a possibility to make it scale [77]; this functions only if the structure of the dataset is extremely simple.

2.3.3 *Summary visualisations and profilers*

A strategy to address the overload is summarisation. When meant for humans, the summaries reconstitute a representative graph, and are limited to the most frequent elements; they are laid out as node-link diagrams, and therefore present only very partial overviews [120, 121, 132]. As for profiler, most of them compute their measures for atomic elements [35], ProLOD [1, 20], LOUPE [76] or AETHER [80], and not for collections; this provide a full and accurate overview, but it is difficult to make sense of it.

In summary, most approaches to browse RDF data from the overview of a collection to the detail of an entity do not scale, due to the very large number of entities in RDF data, each being described by a large number of properties, or paths of properties. Overviews are either barely understandable and usable, or they require much configuration, with little support for navigation.

2.4 OVERVIEWS IN MULTIVARIATE GRAPH VISUALISATION

RDF graphs can be regarded as multivariate graphs. We refer to this survey for an introduction to multivariate graph visualisation [81].

A first approach to consider the question of overviews for multivariate graphs is to ask wether the main visualisation is readable. The main visualizations to represent multiple attributes in multivariate graphs are node-link diagrams and matrices [66, 84, 94]. Node-link diagrams support the display of relatively small and sparse graph. They become difficult to read over a few hundred nodes, especially when the graphs are dense [84, 94]. To a certain extent, optimisation techniques such as edge bundling of force-directed layout algorithms can help gain an overview of the density, even when the details of the diagram are not readable. But with very heterogeneous and dense datasets as in RDF, the result is often a 'big fat graph' [109]. Meanwhile, matrices with advanced zooming and aggregation techniques allow the display of very dense graphs but do not display their structure, especially the paths in the graph [43]. Works combining paths and matrices do it only with very small graphs [12, 112]. Still at a limited scale, a wide range of works show group structures over node-link diagrams and matrices, offering efficient overviews [66, 126].

Another approach is to provide an overview visualisation as a navigation to control the main interface. This overview can be a tree if the data are hierarchical [2], or can be transformed into a hierarchy [8, 51, 94]. Summarised node-link diagrams can also be generated through a more complex analysis of the topology, but are then less automatable [34]. Treemaps can also be used to give an overview of the hierarchy [66]. In general, the problem to automate an overview is that no visualisation seem to fit all graphs [94].

A third approach is to display only one type of attributes, regarded as representative and shared by all nodes in the collection, as we mentioned in the previous section [77]. This allows to apply optimisation techniques such as edge bundling and to meaningfully represent large datasets [93]. To use this approach with several attributes, one could consider the graph as a multilayer network [73]. But this would work only with a few attributes, and again it is not obvious how to automatically select them.

In summary, automating meaningful overviews that can support interactive exploration of large multivariate graphs is still an open issue [94].

3

S-PATHS

This chapter is a revised version of the paper co-authored with Emmanuel Pietriga and Caroline Appert [29]. My contribution included the initial idea, the design of the system, its implementation, its evaluation, and the writing of a significant part of the paper. E. Pietriga and C. Appert supervised the design and contributed writing the paper.

S-Paths is a *mixed-initiative* application [119, 135] designed to support users in the exploration of collections in a dataset, providing them with overviews to help gain insights about those resources. Starting with a view of the whole collection, it offers new overviews as users progress, selecting subsets or asking for different configurations. The tool is available as open-source at gitlab.inria.fr/mdestand/s-paths and can be run online at s-paths.lri.fr.

I first introduce the inspiration for the idea behind *S-Paths*. Then I present the concept of *semantic paths*, to support this idea. After that, I describe the system, explaining how it analyses paths, declares views, and matches them. I explain the user interface in detail, and present a use case on the Nobel Prize dataset. I report on the evaluation we conducted with 9 users. Finally, I discuss limitations and future work.

3.1 INSPIRATION

Starting from the observation that state of the art applications did either produce understandable overviews that were not navigable or navigable overviews that were not understandable, I decided to address the problem by automating the generation of meaningful overviews at any scale.

In La Graphique, Bertin presents a methodology to process raw information to prepare them for efficient visualisations, based on the idea that *"Useful information comes with clusters"* [16]. He builds a matrix with raw data and reorders rows and lines until correlations appear graphically, helping the designer decide which attributes to represent. In his method, a first step consists in simplifying the data: binning, ordering, grouping. This step is automatable. A second step consists in interpreting the data to find higher-level groups, "taking into account [...] the relations with the rest of things". This step requires

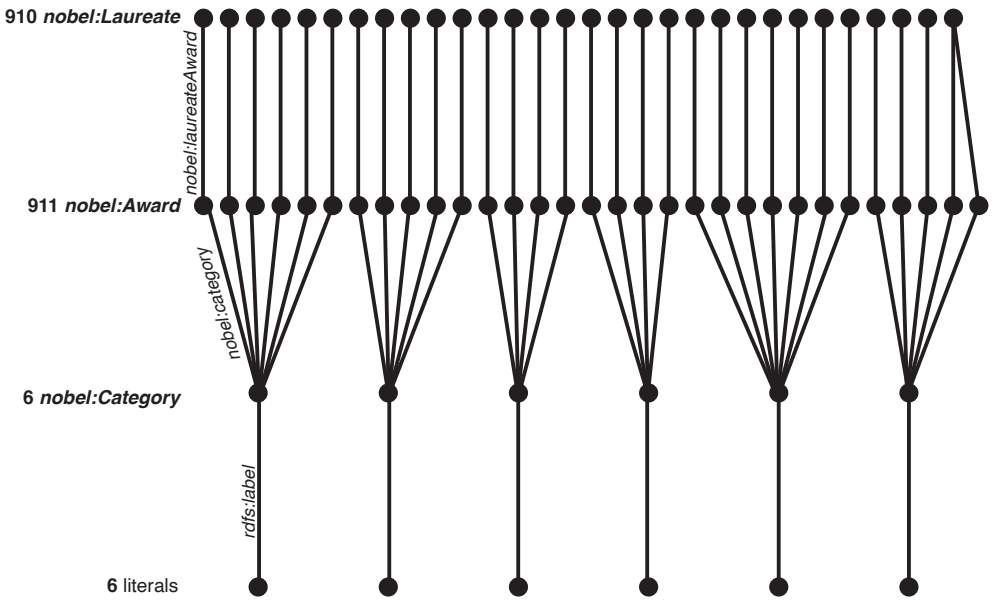


Figure 5: Following the paths to describe a collection of entities allows to'reorganise' its description in higher-level categories.

human intervention. I thought that the relations contained in RDF data could support automating the second step: following the paths in the graph could lead to higher-level groups. For example, from the 910 Nobel laureates in the Nobel RDF dataset, the direct property award would lead to 911 awards, but those awards would then lead to 6 categories, as schematised in Fig. 5. The opposite could also happen, but it should not be too costly to discriminate the paths on the number of unique values at their end. And this could also work if the values were *binnable* literals, with no need to count them. For instance, starting from thousands of documents in a library, the author property would lead to about as many authors, but their birthdate property would allow to group them by decade or century (depending on the total time span).

The third step, the choice of attributes to lay out, relies on identifying correlations visually. To automate this step, computing all correlations did not appear as a reasonable option, given the number of possible combinations. I thought that if each property selected for a visualisation did provide high-level groups, their combination would also provide high-level information. It might not always support the answer to an *overall* question (Fig. 6) but it would at least support the answer to an *average* question, which might even be more efficient to engage users in navigation, letting them identify subsets to inspect.

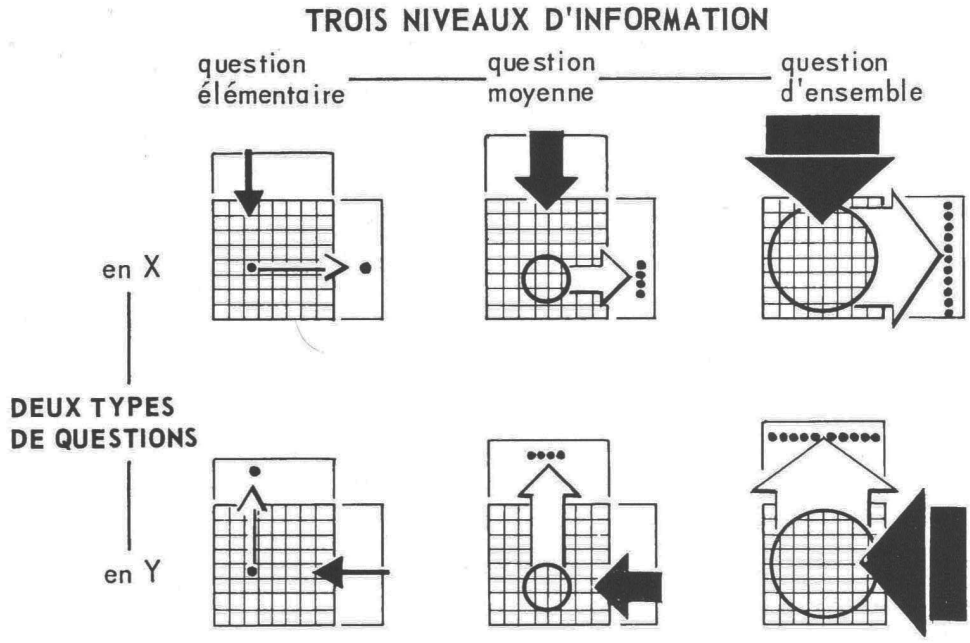


Figure 6: According to Bertin, the efficiency of a visualisation grows with the level of question that it allows to answer.

Therefore, I thought that a system based on a set of templates with high-level visualisations at different scales, and using the datatype and/or number of unique values to discriminate the attributes, should be able to produce overviews at any scale. Early sketches can be found in Appendix A. To support the design of a such a system, we developed the concept of *semantic paths*.

3.2 SEMANTIC PATHS

We introduce the concept of *semantic paths*. We denote U the set URIs and L the set of literals, as is usually done in the literature to define RDF terms. Given an RDF graph G, and a collection E sharing a similarity criterion such that: $\forall e \in \mathrm{E} \subset \mathrm{U}, \exists\ \mathrm{C} \in \mathrm{U}, (e\ \texttt{rdf:type}\ \mathrm{C})$ a semantic path is a set of objects O related to the collection E by a sequence of properties $\mathrm{p}_1, \mathrm{p}_2, \ldots, \mathrm{p}_n$, with $n \geqslant 1$, such that $\exists\ \mathrm{e} \in \mathrm{E}, \exists\ \mathrm{o} \in \mathrm{O} \subset \{\mathrm{U}|\mathrm{L}\}, (e\, \mathrm{p}_1/\mathrm{p}_2/\ldots/\mathrm{p}_n\, \mathrm{o})$.

For a given a collection, *S-Paths* considers all *semantic paths* to identify those that can be matched with its visualisation templates, presented in Table 1. Each template declares the category of paths it is able to display, and the optimal and limit conditions for readability. *S-Paths* uses a set of heuristics to rank the views and displays the best-ranked one by default. Considering

View	Type	Weight	Nb. of resources	dim. 1	dim. 2	dim. 3
density plot	aggregate	0.5		datetime, text or uri	text or uri	
treemap	aggregate	0.3		text or uri		
stacked chart	multiple distinct	0.9	min/max: [2, 1000], optimal: [4, 200]	datetime, text or uri	text or uri	
timeline	multiple distinct	0.85	min/max: [2, 50], optimal: [10, 20]	all datetime paths	text or uri	
URI wheel	multiple distinct	0.4	min: 2	uri		
map	multiple distinct	0.85	min/max: [2,1000]	geo	geo	text
breakdown by values	multiple distinct	0.7	max: 50, optimal: [1, 30]	any category		
images	multiple distinct	0.8	min/max: [2, 1000]	image	text	
info card	single entity	1	min/max: [1,1]	all paths		
node link diagram	single entity	0.5	min/max: [1,1]	all paths		

Table 1: Default configuration of view templates as configured for the Nobel dataset.

paths representing chains of triples, and not only direct properties, gives more opportunities to find properties that are in a displayable range.

We define categories corresponding to different behaviours in terms of aggregation and display, and conceptually roughly equivalent to datatypes. *Datetimes* can be aggregated by years, decades, centuries...; *geographical coordinates* can be aggregated at different scales, as on multi-scale maps; *images* can be resized, displayed at different scales, and juxtaposed in grids or mosaics; *numbers* can be binned; *text strings* can only be aggregated by similar values, and can be displayed according to different layout strategies depending on their length; *URIs* can only be aggregated by similar values. These categories are very similar to the ones used by Atemezing and Troncy [10].

The main characteristics considered are summarized in Table 2. *Depth* captures the degree of indirectness of the property at the end of a path (i. e., the *semantic path* length, or the number of hops to reach it from the original resource). *Completeness* indicates the percentage of entities actually described by the path. This notion of completeness is especially important in

Characteristic	Description
collection	rdf:type shared by the resources
category	one of: `datetime`, `geographical coordinate`, `image`, `number`, `text` or `URI`
depth	number of statements from the set of entities to the set of values
completeness	percentage of entities in the set for which this path actually exists
count	total number of values (or URIs) at the end of the path
unique count	number of unique values (or URIs) at the end of the path

Table 2: Characteristics of *semantic paths*.

the context of semi-structured data, where no schema is enforced, and some properties exist for only a subset of all considered resources.

```
1  SELECT DISTINCT
2      ?p1 ?p2 ?pn                                          | path of depth n
3      (COUNT(DISTINCT ?values) as ?uniqueValues)
4      (COUNT(DISTINCT ?entities) as ?nbCompleteEntities)   | completeness and count
5      (COUNT(?values) as ?totalValues)
6      ?datatype ?isiri ?isliteral ?language                | category
7      (AVG(?charlength) as ?avgcharlength)
8  WHERE {
9      ?entities rdf:type <TYPE_URI> .
10     ?entities ?p1 ?o1 . ?o1 ?p2 ?o2 . ?o2 ?pn ?values .
11     FILTER (?entities != ?o1 &&                          |
12       ?entities != ?o2 &&                                |
13       ?entities != ?values &&                            | prevent from looping
14       ?o1!= ?values &&                                   |
15       ?o2!= ?values &&                                   |
16       ?o1!= ?o2                                          |
17     ) .
18     BIND(datatype(?values) as ?datatype)
19     BIND(ISIRI(?values) AS ?isiri) .
20     BIND(ISLITERAL(?values) AS ?isliteral) .
21     BIND(LANG(?values) AS ?language) .
22     BIND(STRLEN(xsd:string(?values)) AS ?charlength) .
23 }
24 GROUP BY ?p1 ?p2 ?p3 ?datatype ?isiri ?isliteral ?language
```

Figure 7: Conceptual query template to retrieve paths characteristics for all paths of depth *n*. *S-Paths* splits this query into multiple queries, following a divide-and-conquer strategy, as described in Fig. 8.

Path analysis is performed when *S-Paths* gets set up with a new set of graphs, and the characteristics are stored in a Mongo database, which will be retrieved by the system when users navigate. The query to characterize the paths can be formalized as in Fig. 7. However, running such a query is very likely to time out, except with very small datasets. *S-Paths* splits it into multiple queries, following a divide-and-conquer strategy. The analysis function takes as input a set of paths and the similarity criterion for the entities. For each path of maximum length in the set of paths, it queries all extensions by appending one property at the end. Then for each new path found, it queries separately

QUERY TEMPLATE TO RETRIEVE *PATHS EXTENDING* A PATH

```
SELECT DISTINCT ?property WHERE {
    ?entities rdf:type <TYPE_URI> .                                  | given path
    ?entities ?p1 ?o1 . ?o1 ?p2 ?o2 . ?o2 ?pn ?on .
    ?on ?property ?values . | extensions n + 1
    FILTER (?entities != ?o1 &&
      ?entities != ?o2 &&
      ?entities != ?on &&
      ?o1!= ?on &&                                                   | prevent from looping
      ?o2!= ?on &&
      ?o1!= ?o2
    ) .
}
```

QUERY TEMPLATE TO RETRIEVE *COMPLETENESS AND COUNT* FOR A PATH

```
SELECT
    (COUNT(DISTINCT ?values) AS ?unique)                             | completeness and count
    (COUNT(?values) AS ?total)
    (COUNT(DISTINCT ?entities) AS ?nbCompleteEntities)
WHERE {
    ?entities rdf:type <TYPE_URI> .                                  | given path
    ?entities ?p1 ?o1 . ?o1 ?p2 ?o2 . ?o2 ?pn ?values .
    FILTER (?entities != ?o1 &&
      ?entities != ?o2 &&
      ?entities != ?values &&
      ?o1!= ?values &&                                               | prevent from looping
      ?o2!= ?values &&
      ?o1!= ?o2
    ) .
}
```

QUERY TEMPLATE TO RETRIEVE *CATEGORY* FOR A PATH

```
SELECT DISTINCT
    ?datatype ?isiri ?isliteral ?language                            | category
   (AVG(?charlength) as ?avgcharlength)
WHERE {
    ?entities rdf:type <TYPE_URI> .                                  | given path
    ?entities ?p1 ?o1 . ?o1 ?p2 ?o2 . ?o2 ?pn ?values .
    FILTER (?entities != ?o1 &&
      ?entities != ?o2 &&
      ?entities != ?values &&
      ?o1!= ?values &&                                               | prevent from looping
      ?o2!= ?values &&
      ?o1!= ?o2
    ) .
    BIND(datatype(?values) as ?datatype)
    BIND(ISIRI(?values) AS ?isiri) .
    BIND(ISLITERAL(?values) AS ?isliteral) .
    BIND(LANG(?values) AS ?language) .
    BIND(STRLEN(xsd:string(?values)) AS ?charlength) .
}
GROUP BY ?datatype ?isiri ?isliteral ?language
```

Figure 8: Query templates dividing the query in Fig. 7 into multiple queries. The first query is called iteratively for each path of depth n, to find all the paths of depth $n+1$ extending this path. The second and third queries are called for each path.

for completeness and counts, and datatype information, as shown in Fig. 8. Then the analysis function saves its results and calls itself recursively until the maximum length of paths set in the configuration is reached. This allows

to progress in the exploration step by step, and to resume where it stopped if the network or SPARQL endpoint breaks.

3.3 SYSTEM

3.3.1 *View specification*

S-Paths provides a set of views: map, image gallery, timeline, statistical charts, simple node-link diagrams, presented in Table 1. Each view specifies how many dimensions it can handle, and defines the requirements that semantic paths must meet to be considered for a given dimension, as well as the conditions under which they will be considered optimal. For example, Fig. 9 shows the definition of a binned scatterplot view. It handles two dimensions: x- and y-axis. Paths associated with the x-axis (dimension 1) must be of *category* `datetime`, `text` or `URI`. If the category is `datetime`, the number of distinct values (*unique count*) is not limited as they can be aggregated meaningfully (line 8). On the contrary, if the category is `text`, the *unique count* must not exceed 150 (line 9).

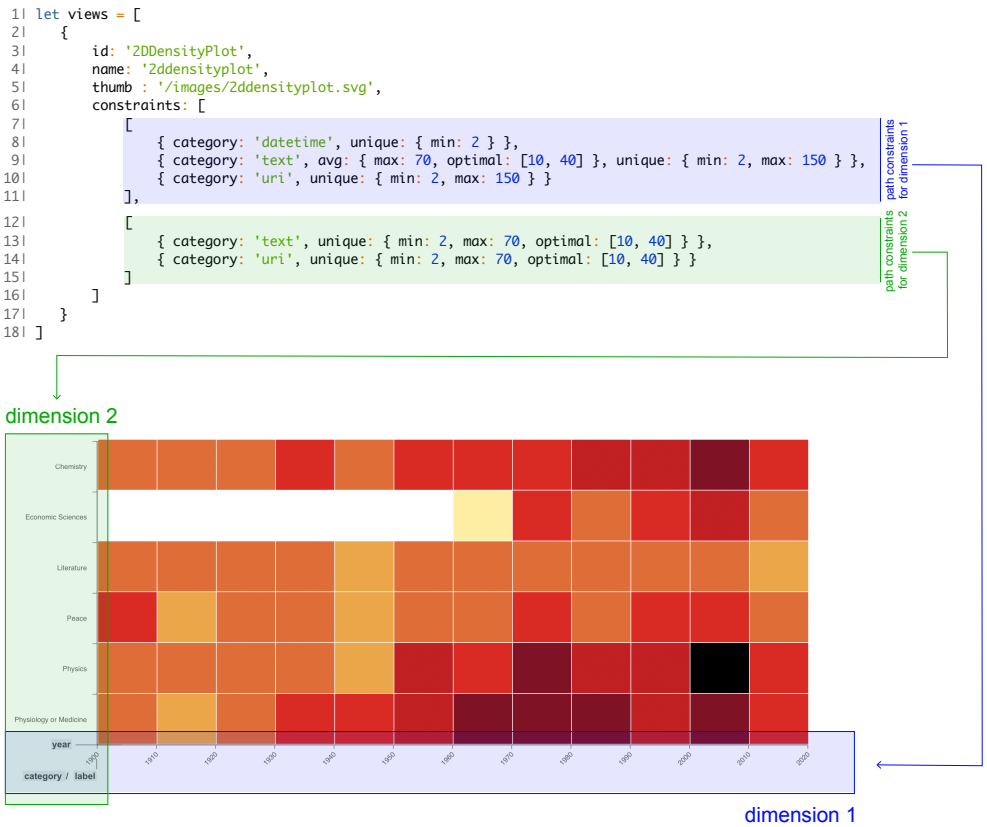


Figure 9: Binned scatterplot view definition. The view defines 2 dimensions, the first dimension is mapped to the x axis and the second to the y axis.

AGGREGATE VIEWS

```
SELECT DISTINCT
    COUNT(DISTINCT ?values_dimension_1)
    ?values_dimension_1
    COUNT(DISTINCT ?values_dimension_2)
    ?values_dimension_2
    COUNT(DISTINCT ?values_dimension_n)
    ?values_dimension_n
WHERE {
    ?entities rdf:type <TYPE_URI> .

    ?entities prevp1/prevp2/prevpn ?setconstraints .
    FILTER EXP where exp is an expression of SPARQL filter language*.

    ?entities p1/p2/pn ?values_dimension_1 .
    ?entities p3/p4/pn ?bind_values_dimension_2 .
    BIND(FLOOR(?bind_values_dimension_1/?coef)*?coef as ?values_dimension_2)}
    ?entities p5/p6/pn ?values_dimension_n .
}
GROUP BY
    ?values_dimension_1
    ?values_dimension_2
    ?values_dimension_n
```

combined aggregates of the values at the end of paths for the n dimensions in the view

collection

subset defined by user selections in previous views

VIEWS DISPLAYING SEVERAL ENTITIES WITHOUT AGGREGATION

```
SELECT DISTINCT
    ?entities
    ?values_dimension_1
    ?values_dimension_2
    ?values_dimension_n
WHERE {
    ?entities rdf:type <TYPE_URI> .

    ?entities prevp1/prevp2/prevpn ?setconstraints .
    FILTER EXP where exp is an expression of SPARQL filter language*.

    ?entities p1/p2/pn ?values_dimension_1 .
    ?entities p3/p4/pn ?values_dimension_2 .
    ?entities p5/p6/pn ?values_dimension_n .
}
```

for each entity values at the end of paths for the n dimensions in the view

collection

subset defined by user selections in previous views

SINGLE ENTITY VIEWS

```
SELECT
    ?p1
    ?p2
    ?pn
    ?values
WHERE {
    { <SINGLE_ENTITY_URI> p1 ?values . }
    UNION
    { <SINGLE_ENTITY_URI> p1/p2 ?values . }
    UNION
    { <SINGLE_ENTITY_URI> p1/p2/pn ?values . }
}
```

all the paths and values at their end for a single entity

*https://www.w3.org/TR/sparql11-query/#rConstraint

Figure 10: Query templates used to populate views. The 3 types of queries correspond to the view type listed in Table 1.

Each view also receives a weight. Generic views—that are able to handle any data—can be lower-rated, while very specific views able to give a more meaningful representation but only for specific types of data can be given higher priority. Petrelli *et al.* [92] refer to space, time and topology as exam-

ples of *graspable* dimensions that provide effective support to the exploration and sense-making of semantic data. Indeed, the aggregation mechanism is embedded in the map background and in our mind, which explains why they scale so well: when we look at a world map, we automatically think in terms of continents, while when we look at a zoomed map, we think in terms of countries or region. Our mind seems to be continuously looking for the most graspable dimension.

This also makes it possible to have various levels of overview, depending on the number of items in the selection when dealing with very large sets. The system should support, and default to, aggregate views on the data, enabling users to easily select subsets of higher interest to focus on, eventually displaying them in detail [114] when the size of the subset becomes tractable.

3.3.2 *Matching algorithm*

Once semantic paths for a given collection have been retrieved, the system evaluates the suitability of the different views to generate a default representation of this collection. Fig. 11 gives an overview of the process. *S-Paths* iterates through the entire collection of views, discards the ones that are not viable for the collection considered, configures the remaining ones with the top-ranked semantic paths as dimensions, and gives a score to each candidate view, eventually selecting the top-ranked one.

Some views define the maximum number of resources they can handle. When the number of resources to visualise exceeds that maximum number, this view is discarded. Otherwise, *S-Paths* computes the list of candidate paths for each of the view's dimensions. If there is no path matching the constraints for one of the dimensions, the view is discarded.

S-Paths assigns a score to each path, using a normalised weighted average of the following criteria:

- ***path completeness***: *S-Paths* favours paths that have a high level of support (cover a large number of resources);
- ***adequation between the path and the view dimension***: a path closer to the optimal settings for a dimension is scored higher. A dimension can specify optimal settings in terms of *unique count* (Fig. 9, lines 8-10 and 13-14), and in terms of average char length (*avgcharlength*) for paths leading to textual values. Also, the closer to the ranges defined as optimal for the dimension, the better the score. When a dimension supports several *categories*, it lists those categories in order of preference. This preference

influences the score. For example, in Fig. 9, `datetime` is preferred over `text` for the x-axis, which is itself preferred over `uri`.

- ***path depth***: *S-Paths* favors more direct properties (lower depth) over more indirect ones (higher depth);
- ***custom path preferences***: optionally, *S-Paths* can be told to favor some paths, by declaring them explicitly in a configuration file.

After having iterated over the collection of views, *S-Paths* builds a list of optimally-configured views, retaining only the best-scoring paths for each view (Fig. 11, 1st row 3rd column). It then assigns a score to each view using another normalised weighted average of:

- ***configuration quality***: the average of associated path scores;
- ***preference***: each type of view has a score which indicates preferences for some types over others based on, e. g., their familiarity or concreteness. As this is subjective and application domain-dependent, these scores can be edited in a configuration file;
- ***number of dimensions***: support for more dimensions to be displayed simultaneously implies more opportunities for visualising different properties.

S-Paths then selects the top-scoring view according to this weighted average, and configures it with the top-ranked semantic paths (Fig. 11, 2nd row). Lower-ranked paths that still match this view can be selected using the *dimensions* menus. The *view* menu lets users switch to any other view available for the collection.

When users select a subset, the system retrieves paths for the entire collection, and computes an approximation for the subset.

3.3.3 *SPARQL query mechanism*

S-Paths' query mechanism is independent of any view specifics. It switches between three possible query templates according to the type of view – *aggregate view*, *multiple distinct entities*, or *single entity* – as detailed in Fig. 10. In the case of aggregate views, binning operations are performed in the query for paths belonging to *datetime*, *geo* or *number* category, so as to ensure the number of results will not exceed the endpoint quota (lines 15-16). Queries which handle transitions between views are identical, combining dimensions from the previous and the new view.

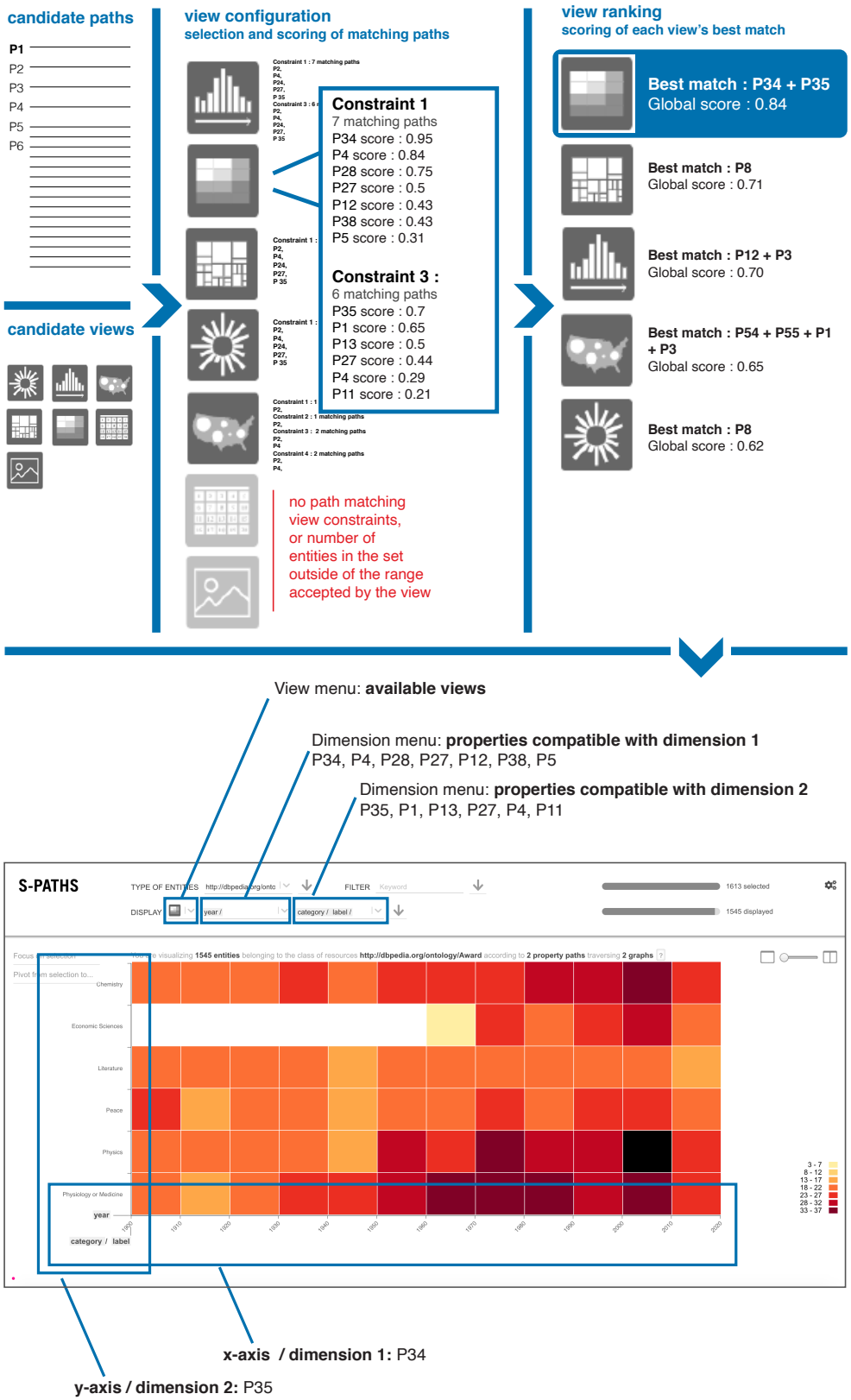


Figure 11: Process for generating a default view. The matching algorithm compares the *semantic paths* with the view requirements to present the most readable overview to users.

When a subset is selected in an *aggregate view*, constraints to define the subset (lines 11-12 or 31-32) are added to the ones already gathered in previous views. When the selection happens in a *multiple distinct entities* view, previous constraints are replaced by the list of selected URIs. When users *pivot* to explore another collection, constraints defined in previous views are rewritten: `?entities` is renamed `?oldentities`, and the relation between `?entities` and `?oldentities` is added.

3.3.4 *Configuration*

S-Paths requires minimal configuration: the URIs of the SPARQL endpoint and of the named graphs to explore behind it. At the start, the system looks for `rdf:type` statements in those graphs, and displays a list of classes which can be analyzed. A configuration file enables to adjust other optional parameters, in order to adapt the system to a specific dataset: the maximum length of the paths, the weight of the different criteria used in the matching algorithm, and the prefixes (namespace bindings) for URIs.

3.3.5 *Implementation*

S-Paths is developed using NodeJS/Express. RDF data is stored in a Virtuoso instance, and queried using SPARQL. Semantic paths are encoded using the Fresnel FSL syntax [96]. The code analyzing them runs server-side, storing their characteristics in MongoDB.

When populating a view, *S-Paths* only fetches the data actually displayed in the view, so as to improve scalability and support browsing large collections. It queries the SPARQL endpoint using query patterns that retrieve only the distinct values at the end of the property paths.

The front-end is implemented as a Single Page Application developed with React and Redux, a React component being associated with each view. Views are implemented in a modular way, so as to ease the process of extending *S-Paths* with new types of views. New views have to implement a minimal interface to fit in the general framework: publish their constraints on allowed dimensions; broadcast subset selections made by users in the view; and broadcast graphical properties of the entities they display to enable interpolated graphics transitions, as well as brushing & linking between views. Beyond this, each view is free to handle the data in its own way: clustering and other aggregation methods, libraries used for graphics rendering (many views are implemented with Vega [106], others are plain HTML+CSS), *etc.*

S-Paths is distributed as an open-source project[1]. The demo instance[2] runs on a Linux virtual machine with 4 vCPUs, 16GB RAM, 32 GB disk space. This demo currently runs on a single Virtuoso instance, using the default configuration.

3.4 USER INTERACTION

The system then provides them with different options as entry points into the data, which correspond to the different collections detected (classes of resources). By default, it selects the class of resources that has the richest description and generates a default view that acts as a gateway to that collection.

Fig. 22 shows the interface when starting to browse the earlier-mentioned Nobel prize dataset. It is composed of a set of widgets in the upper part, and a central panel showing a visualisation of the collection. *S-Paths* has selected the `Award` class of resources as the entry point. The collection containing more than 1,500 awards, the system defaults to an aggregate visualisation (the colour of individual cells encodes their element count), selecting the award's `year` and its `category` as the two dimensions to display.

S-Paths displays summary information about the collection and the visualised semantic paths just above the view itself, providing users with some context (see, e. g., Fig. 22). Detailed information about the selected semantic paths and the RDF graphs they traverse is also available. Such provenance-related information is especially useful when dealing with heterogeneous datasets distributed over multiple linked graphs, but as it is rather expert-oriented, it gets displayed on-demand only, by clicking an icon [?] next to the summary.

From a presentation perspective, *S-Paths* seeks to display user-friendly labels [33] for the resources and properties encountered along semantic paths. The corresponding URIs are systematically dereferenced, looking for labels and descriptions (`rdfs:label`, *etc.*). However, its visualisations still allow making sense of URIs when no label is available. For instance, Fig. 12

These are used in the interface: next to the resource selection menu, in the view configuration menu, in the axes' legends, and whenever a semantic path is displayed in a view template. When no label can be found, the system falls back to the prefixed URI using the prefix.cc Web service.

1 https://gitlab.inria.fr/mdestand/s-paths
2 http://s-paths.lri.fr

Figure 12: Example views for the French National Library data, showing `foaf:Document` resources along two semantic paths: `foaf:Document/dcterms:subject/*/foaf:focus/*/bnf:languageOfThePerson/*` and `foaf:Document/dcterms:subject/*/bnf:dateCreated/*`. Although no path with a label was found for the y axis, we have a simple, synthetic view of documents organized according to their author's date of birth, and language. A use case for it could be librarians needing to identify interesting authors and documents to prepare a cultural diplomacy event honoring a particular culture.

3.4.1 *Selection of a collection*

Users can select another collection in the top row of the UI. They can switch to any other class available in the COLLECTION menu. Available collections correspond to the sets of entities that share a particular *class of resources* (`rdf:type` statements. Fig. 13 shows the default view when the `nobel:Laureate` collection is selected. Users can select a subset by specifying a FILTER in the form of keywords to be matched in values anywhere along the associated semantic paths. They can use such a filter to restrict the initial `Laureate` set (910 resources) to laureates related to keyword `medicine` (217 resources).

3.4.2 *View configuration*

The drop-down menus in the second row of the UI (*dimensions* menus) let users change which semantic paths get visualised in the view. They support auto-completion for quick selection in the list. The number of *dimensions* menus depends on how many dimensions the view expects. For instance, the default view generated by *S-Paths* for the `Award` class is a histogram (Fig. 13) showing the distribution by award year (horizontal axis) and by gender (colour). The user can change the semantic paths set in the two *dimensions* menu. In Fig. 14, the histogram has been reconfigured to show the

Figure 13: Default view on the `nobel:Laureate` collection, showing their date of birth (*semantic path* of depth 1 `nobel:Laureate/dbpedia:dateOfBirth/*` mapped to x-axis) and their gender (*semantic path* of depth 1 `nobel:Laureate/foaf:gender/*` mapped to colour scale).

Figure 14: Another view on the `nobel:Laureate` collection, switching dimensions to award year (*semantic path* of depth 2 `nobel:Laureate/nobel:nobelPrize/*/nobel:year/*` aggregated by decade) and discipline (*semantic path* of depth 3 `nobel:Laureate/nobel:laureateAward/*/nobel:category/*/rdfs:label/*` mapped to colour scale).

distribution by birth date and by category. Each menu only features paths whose values are of a type compatible with the associated dimension in the chart. For instance, on a map view, the first two menus, which correspond to the latitude and longitude of items to be plotted, will only feature semantic paths that point to geo-location properties such as, e. g., `wgs84_pos:lat` and

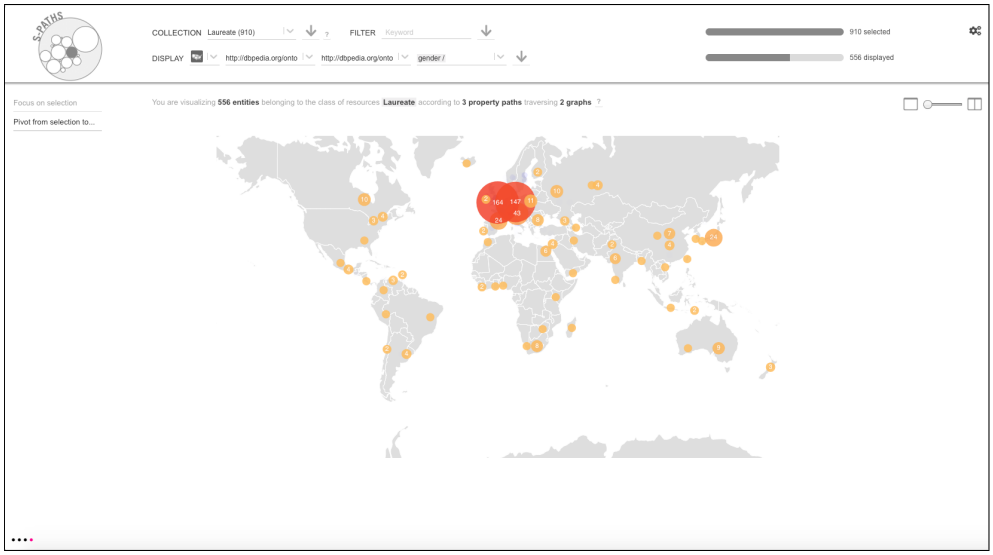


Figure 15: Another view on `nobel:Laureate` collection, switching to a map view offers as first choice the laureates' birthplace latitude and longitude as the dimensions used to plot Nobel laureates on the map (*semantic path* of depth 3 `nobel:Laureate/dbpedia:birthPlace/*/owl:sameAs/*/wgs84_pos:lat/*` and `wgs84_pos:long/*`). It is possible to switch to the `dbpedia:deathPlace`.

`wgs84_pos:long`. Similarly, the timeline view will only allow semantic paths ending with time-related properties for its first dimension.

Users can also choose another type of visualisation using the *view* menu , which lists all visualisations compatible with the current collection set or subset (chart, timeline, map, image gallery, *etc.*). Selecting a different visualisation in this menu will generate a new view based on the top-ranked semantic paths for that view. For example, in Fig. 15, the user has selected the map view. *S-Paths* automatically populates this view with longer *semantic paths* of depth 3: `dbpedia:birthPlace/owl:sameAs/wgs84_pos:lat`, `dbpedia:birthPlace/owl:sameAs/wgs84_pos:long`.

3.4.3 *Subset selection*

The above view reconfiguration capabilities let users change *what dimensions* of resources in the current set are visualised, and *how* they are visualised. Users can also restrict *what resources* to visualise by making direct selections in the currently displayed view: clicking on individual items and aggregates, performing rubber-band selections of contiguous elements, selecting ranges by, e. g., clicking a particular bin on the horizontal axis of a histogram to select all items in that bar. They can also combine multiple, non-contiguous selections by holding a modifier key (Shift), as in popular graphics-oriented applications such as presentation programs and graphics editors. Once such a

sub-selection has been made, users can turn it into the new collection to explore. The process can be repeated iteratively. Combined with the automatic aggregation of resources along the chosen dimensions, which only occurs when the collection is too large, this selection mechanism provides users with means to effectively zoom-in on part of the data and get details on demand [114]. For example, starting from the map in Fig. 15, it is possible to select laureates in South America and Africa only, and focus on them.

The two percentage bars in the top right corner of the interface give an indication of what proportion of the dataset is currently visualised. The upper bar indicates how many resources of the selected type match the FILTER. The lower bar indicates how many resources in the collection are actually represented in the main view. This latter collection depends on the successive user selections, and on the selected *semantic paths*, that might not exist for all resources in the current collection.

3.4.4 *Navigation and transitions between views*

S-Paths smoothly animates transitions between views when the two views have entities in common [47]. This provides some basic level of perceptual continuity that contributes to minimizing the cognitive cost of relating one view to the next, as illustrated in Fig. 16, Fig. 17, Fig. 18, Fig. 19 and Fig. 20.

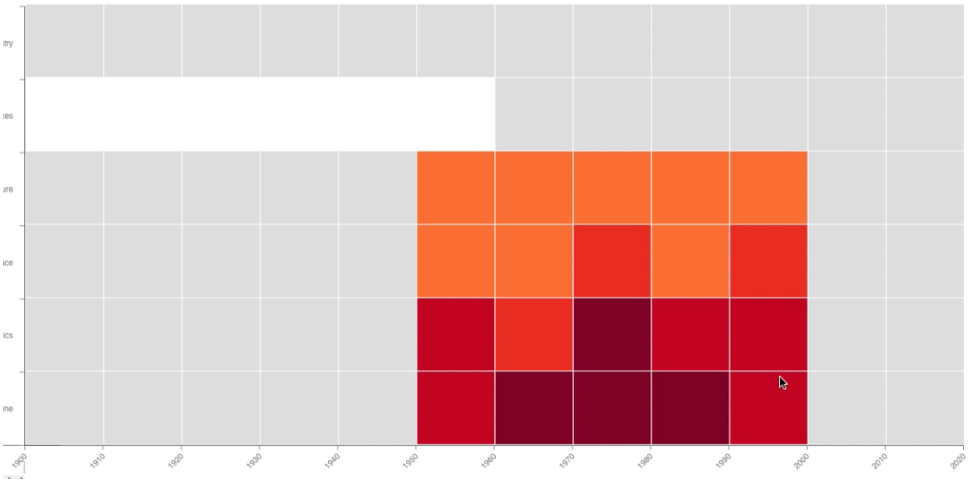


Figure 16: Animated transition from a sub-selection made in a binned scatterplot showing counts for `nobel:AwardFile` aggregated by decade, to a histogram showing the distribution of prizes per discipline for each individual year. Start state.

Visual marks that represent aggregations of resources are partitioned according to a space-filling strategy. This preserves aggregations that still exist in the target view and might actually be part of a larger aggregate.

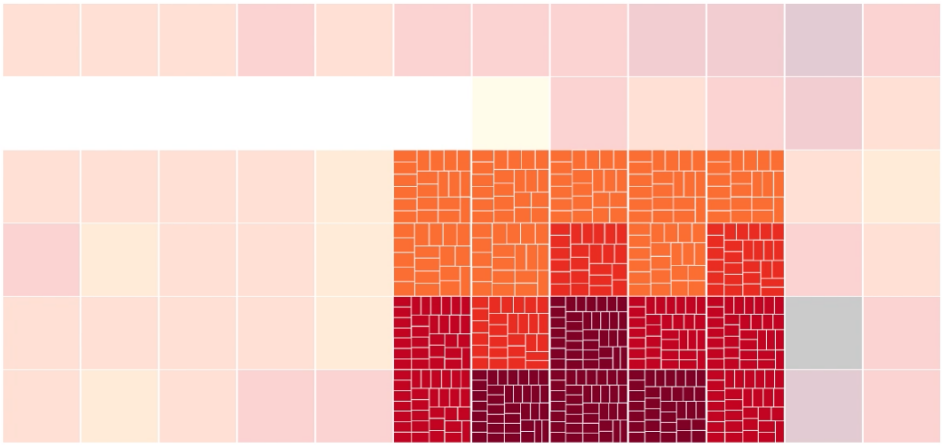


Figure 17: Animated transition, sample intermediate frame. Elements no longer present in the target view are faded out, and visual marks corresponding to aggregates get decomposed into groups that will transition towards the same target.

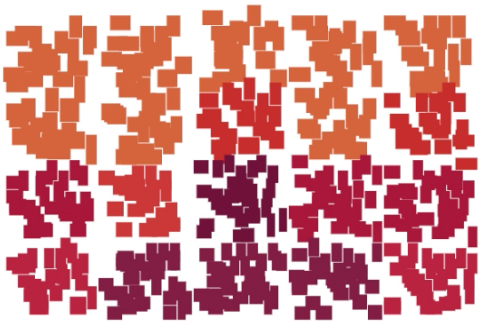


Figure 18: Animated transition, sample intermediate frame. Groups get smoothly interpolated along relevant encoding channels: position, colour, shape.

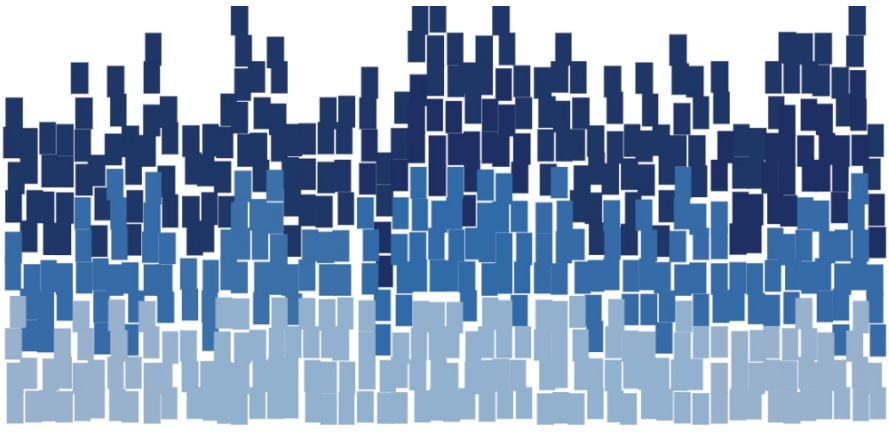


Figure 19: Animated transition, sample intermediate frame. Elements that did not exist in the source view fade in.

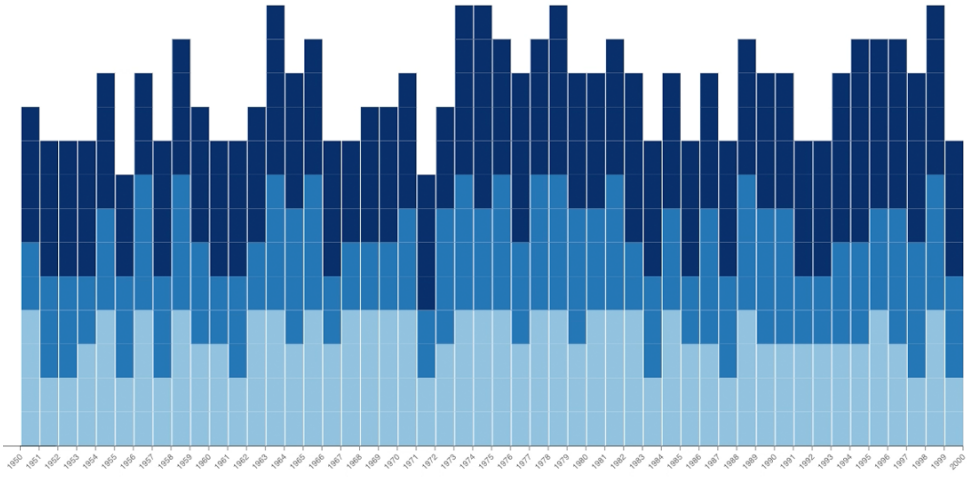


Figure 20: Animated transition, end state.

The system also supports the juxtaposition of two consecutive views, as well as brushing and linking between those views: selecting elements in one view immediately highlights the corresponding elements in the other view (see Fig. 21), further helping users relate views. The same space-filling strategy as above is used to handle brushing & linking between aggregates.

S-Paths also keeps track of all past views and represents them as dots forming a basic navigation history displayed in the bottom left corner of the interface. Clicking on one of these dots reverts to the corresponding view, enabling users to easily backtrack.

Transitioning from one view to another requires an explicit user action in *S-Paths*. Users have to click button Focus on selection to focus on a subset selection, or one of the ↓ buttons to apply other configuration changes. While an explicit user action is necessary to offer the multiple-selection model described above, configuration actions would preferably not require users to explicitly apply changes, in order to promote responsiveness and immediate feedback. But, as changes to the collection and to the view settings can take several seconds (depending on the complexity of the underlying SPARQL queries, on the size of the collection considered, and on the overall responsiveness of the queried endpoints), this is impossible in practice. Triple store query performance needs to improve by at least an order of magnitude before this can be seriously considered.

3.4.5 *Pivot*

Switching to another collection related to the current set is an essential operation in set-based navigation. The classic example of such a pivot opera-

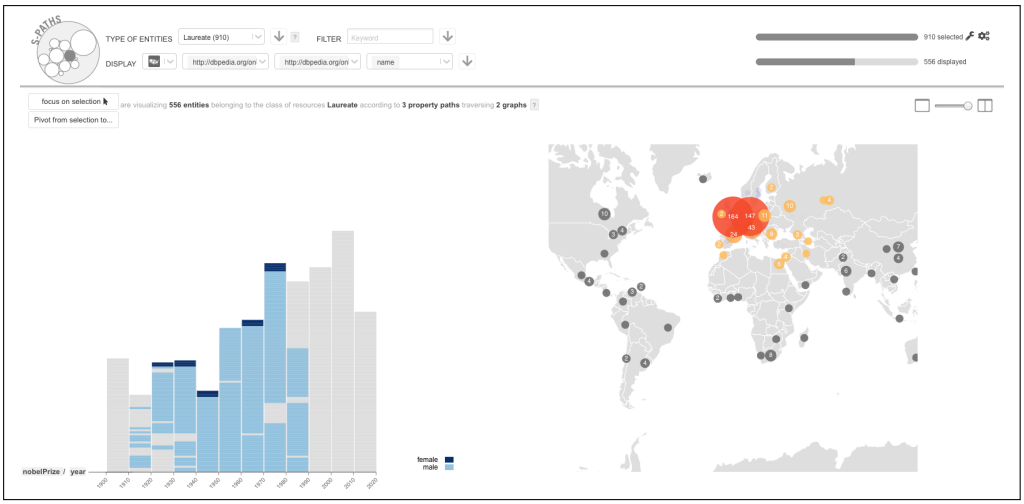


Figure 21: Brushing & Linking between two views: items selected on the map are highlighted in the histogram.

tion can be found in Parallax [59] where focus switches from *all presidents of the USA* to *the children of presidents affiliated with the Republican party*. In *S-Paths*, clicking Pivot from selection to... lists all possibilities for pivoting from the current view. These include *root pivots*, and *path pivots*. Root pivots are other classes that entities in the current collection itself may belong to (multiple `rdf:type` values). For instance, in Fig. 22, `nobel:LaureateAward` and `nobel:NobelPrize` are listed as options for pivoting, as some entities in the current collection (`dbpedia-owl:Award`) also belong to one or both of those classes. Choosing e. g., the `nobel:NobelPrize` pivot will select the collection from the current view that belong to the `nobel:NobelPrize` class. Path pivots are collections found on the *semantic paths* used as dimensions in the view. For example, path `nobel:category/rdfs:label` is used as the second dimension in the view (Fig. 22); `nobel:Category` thus gets listed as the third option for pivoting. In all cases, pivoting takes into account subselections made by users in the view. For instance, selecting the column corresponding to the 1940's in Fig. 22 and pivoting to `Category` would lead to a view showing only 5 out of all 6 categories, as the prize in `Economic Sciences` was established in 1968.

3.5 ILLUSTRATIVE SCENARIO

This section illustrates how *S-Paths* works using a simple scenario. We follow Alice, a lay user interested in the data made available by the Nobel Prize organization. We will see how *S-Paths* enables her to find interesting facts. The

first screen displayed by *S-Paths* is a binned scatterplot showing information about `Awards`: their `category` and `year` (Fig. 22).

Alice notices that five of the six categories start at the beginning of the 20th century, while the sixth, Economic Sciences, was only created in the 1960's. From the colours in the cells, she sees that disciplines such as Physics and Medicine have more awards than, e. g., Literature. She wonders if she can find an explanation, which prompts her to try other dimensions than `year` for the horizontal axis.

Using the first *dimension* menu, Alice considers different options. Hovering the different property paths, she can read their description. She finds the `share` property (Fig. 22), which corresponds to the number of shares a prize is divided into. She selects it. This displays the view in Fig. 23, which reveals that awards in the `Literature` category (3rd row from top) tend to have a lower `share` value (often 1, sometimes 2) compared to those in the other categories (more evenly distributed between 1 and 4). She now understands why the number of `Awards` is higher in scientific disciplines: prizes are mostly given to two-to-four people in the latter case, while they are often given to a single person in `Literature`.

Alice would like to know more about `Laureates` in `Literature`. She selects that line in the view and asks *S-Paths* to focus on it. Looking at the resulting stacked chart, she is struck by the unbalanced distribution between men and women (Fig. 24). She reconfigures the view to get a map, displaying the geographical origin of `Laureates` (Fig. 25). She notices that many of them were born in Europe.

Alice wonders if gender repartition is similar in all categories. Using the top menu, she consider the entire `Laureates` collection (Fig. 26). Her attention is caught by the fact that there is only one `Laureate` born in the 1990's and that she is a woman. Alice selects this single entity and focuses on it. The generic info card (Fig. 27) detailing this `Laureate` resource reveals that this is Malala Yousafzai, a young Pakistani who was awarded the peace prize in 2014.

Coming back to her original question about gender repartition, Alice goes back to the previous screen (Fig. 26), which she reconfigures to display gender and category (Fig. 28). She observes that gender unbalance is particularly pronounced in `Chemistry`, `Economic Sciences`, and `Physics`. She selects female laureates and focuses on them. This yields a timeline view (Fig. 29) in which she can read their name, as well as their birth, death and award dates.

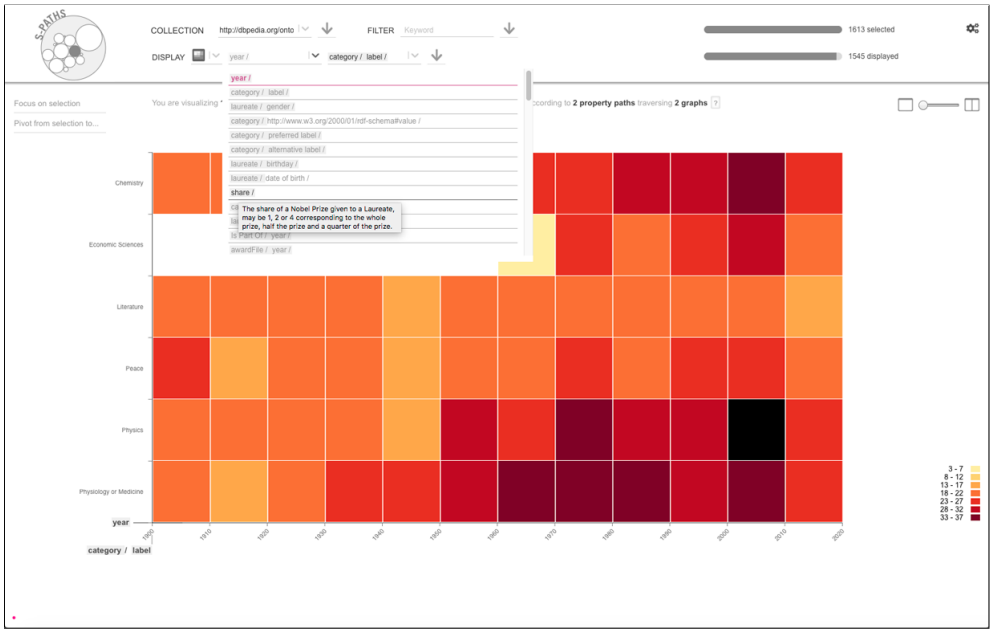


Figure 22: Nobel prize use case (a) binned scatterplot showing the count of `Awards` per year (`dbpedia:Award/nobel:year/*`, binned by decade) and category (`dbpedia:Award/nobel:category/*`).

Figure 23: Nobel prize use case (b): binned scatterplot showing the count of `Awards` per award share (`dbpedia:Award/nobel:share/*`) and category (`dbpedia:Award/nobel:category/*`).

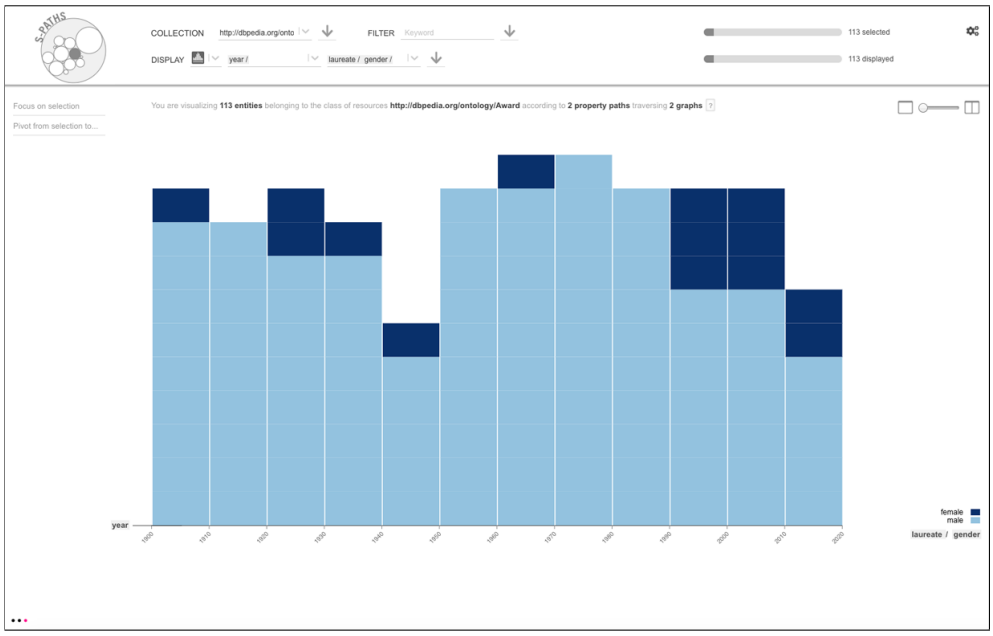


Figure 24: Nobel prize use case: (c) histogram showing the repartition of `Awards` over the years (`dbpedia:Award/nobel:year/*`) by gender (`dbpedia:Award/nobel:laureate/*/gender/*`).

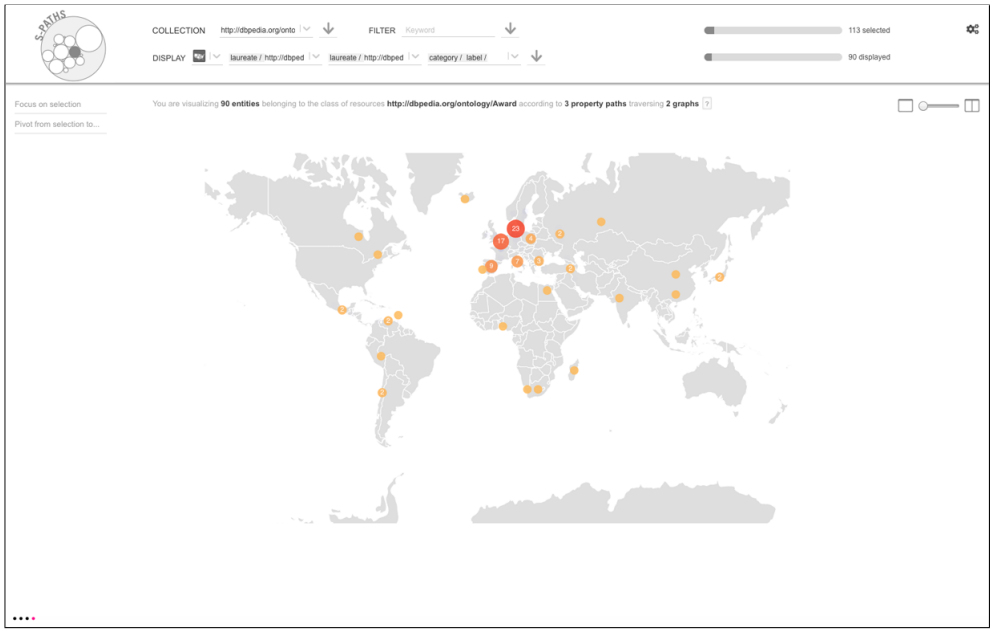


Figure 25: Nobel prize use case: (d) map showing `Awards` along latitude (`dbpedia:Award/dbpedia:birthPlace/*/owl:sameAs/*/wgs84_pos:lat/*`) and longitude (`dbpedia:Award/dbpedia:birthPlace/*/owl:sameAs/*/wgs84_pos:long/*`).

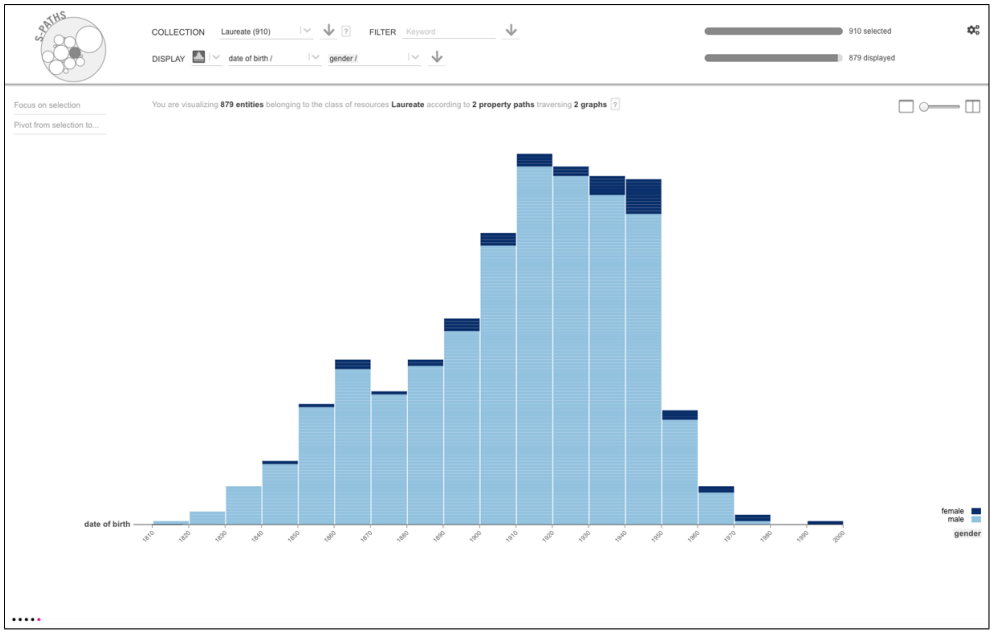


Figure 26: Nobel prize use case: (e) histogram showing `Laureates` by gender (`nobel:Laureate/foaf:gender/*`) and date of birth (`nobel:Laureate/dbpedia:dateOfBirth/*`).

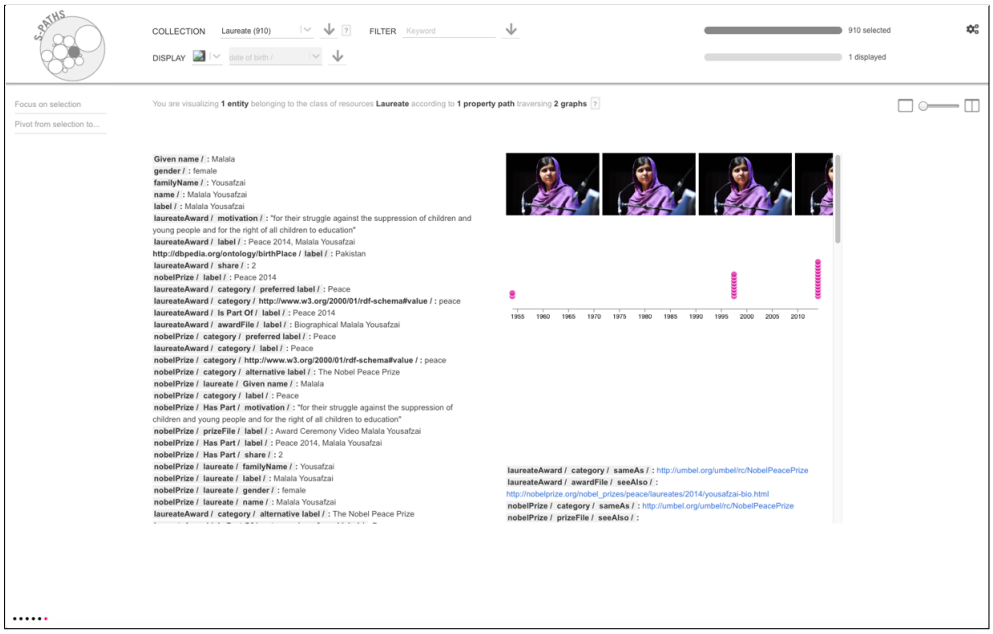


Figure 27: Nobel prize use case: (f) info card detailing all *semantic paths* for one laureate in the collection.

Figure 28: Nobel prize use case: (g) histogram showing `Laureates` by gender (`nobel:Laureate/foaf:gender/*`) balanced by category (`nobel:Laureate/nobel:category/*`).

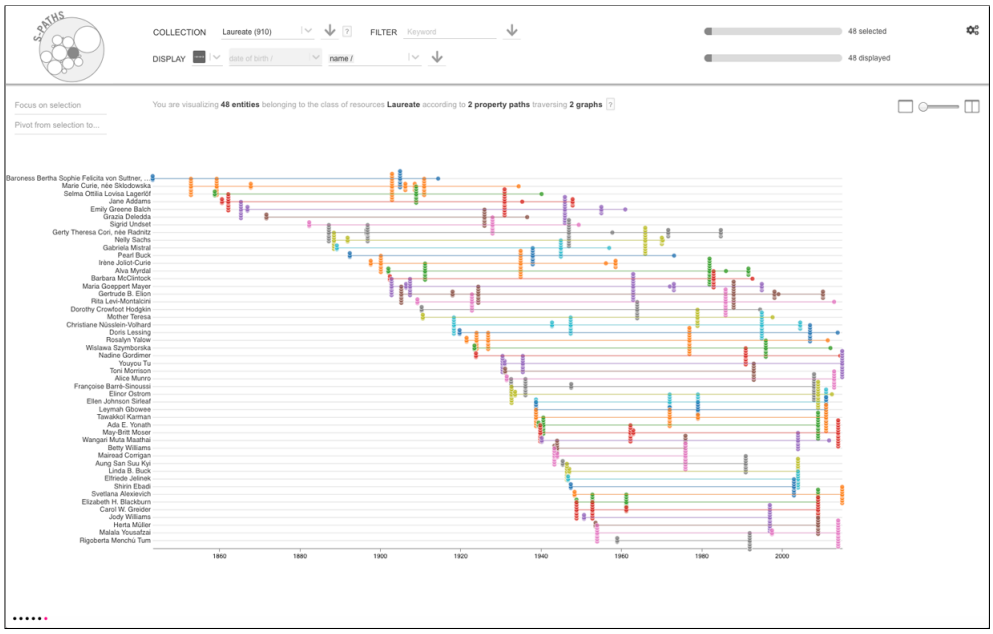


Figure 29: Nobel prize use case: (h) timeline of all events for a subset of `nobel:Laureate`.

3.6 EVALUATION

At the end of *S-Paths*' design process, we conducted a qualitative study to observe how *S-Paths* supports users in their exploration of a dataset. The study consisted of a series of nine individual sessions: 2 *data producers*, 2 *data reusers* and 5 *lay users*. Each session started with the operator demonstrating *S-Paths* for ten minutes. Participants then explored a dataset for twenty minutes and were encouraged to *think aloud*. Sessions ended with a questionnaire to gather feedback about their experience. Each session lasted an hour.

The exploratory tasks to be performed depended on the persona. Lay users were provided with a list of tasks in order not to get them inhibited by the open-ended nature of the task, but the operator made it clear that tasks were indicative, and that participants were free to explore the dataset as they wanted. Reusers were asked to discover a dataset as if they were in a hackathon: explore a dataset and find a potential application to develop based on these data. Sessions with both lay users and reusers were conducted with the Nobel prize dataset. Data producers were asked to take a fresh look at their own data.

Our hypothesis was that aggregating data along readable dimensions as *S-Paths* does would help users gather high-level knowledge about a dataset. In particular, *S-Paths* helps spot outliers (e. g., there is only one laureate born in the 1990's, and she is a woman), but also shows categories or trends (e. g., awards in literature are usually not shared). In the rest of this section, we report on our observations along four axes that White & Roth [133] identify as key aspects for the evaluation of an exploratory search system: learning and cognition, information novelty, engagement and enjoyment. We do not analyse task success and task time, the number of participants being too low to yield significant results.

3.6.1 *Learning and cognition*

In order to assess how much users learnt during their use of *S-Paths*, *reusers* and *lay users* had to answer a series of five questions about Nobel prizes in the final questionnaire.[3] They also had to tell whether they would have been able to answer those questions before the experiment. All seven participants successfully answered the series of five questions, and would not have been able to do so before the experiment in most cases (1 reuser knew the answer for 3 questions, 1 lay user for 1 question, all 5 other participants knew none

3 Questions for lay users and reusers are in Appendix C.

Figure 30: Participants to our evaluation, corresponding to our 3 persona: *data producers*, *data reusers* and *lay users*.

of the answers). The questionnaire additionally asked participants to report other facts they had learnt. They all reported learning some facts, ranging from specific ones (e. g., “Marie Curie is the first woman to get an award in 1903” or “Vernon Smith was awarded the only prize in *economic psychology*”) to general ones (e. g., “Many Nobel prizes were born in the UK” or “Organizations can be awarded a prize”).

As participants were thinking aloud, the operator noticed that it was not always clear to lay users which entity was represented in a view. However, it did not seem to have much impact on their understanding. For example, they might not have known that the view in Fig. 22 was showing the number of Laureate Awards and what the definition of a Laureate Award was. However, it did not prevent them from learning that there were six categories for Nobel prize, and that those awards have existed for over a century.

3.6.2 *Information novelty*

S-Paths makes it easy for users to get views that combine several paths, providing insights that would otherwise have required some analysis. Data reusers and data producers were the most enthusiastic about getting such combined views that reveal distributions, trends or outliers in the data. One reuser mentioned that *S-Paths* automatically does what he typically does by means of multiple SPARQL queries or Python scripts to analyze dumps at the start of a hackathon: “*When you engage in a hackathon, what you are looking for is the irregularity in the data, and this tool finds them and points them out.*”.

The two data producers spontaneously identified specific cases where *S-Paths* may be very useful. The first data publisher is in charge of the ontology of Legilux,[4] a dataset containing all of Luxembourg’s legal texts. He often gets requests for data from the Luxembourg publications office, whose experts need to answer questions coming from the government; or to react to statements made by journalists. Providing those data requires writing SPARQL queries, and importing their result sets into a spreadsheet to generate charts. *S-Paths* would remove the need to write queries and resort to other tools for data presentation, and would thus greatly improve such a workflow.

The second data publisher is the Bibliothèque nationale de France (BnF). Their data come from multiple catalogues, and are enriched with external data, making it particularly difficult to get an overview of their dataset. They liked the fact that *S-Paths* provides views that combine metadata with meta-

4 http://legilux.public.lu/

metadata (e. g., a view of works that combines the work's topic with the year when the work was added to the catalogue). This helps understand the cataloguing policy and trends and spot anomalies in the data. In particular, *S-Paths* could drive campaigns for cleaning and fixing data.

3.6.3 *Engagement and enjoyment*

All participants were fully focused on their tasks. Twenty minutes seemed short to them. Lay users were able to use the tool without knowing about graph databases or what a path in a graph is. However, some of them expressed concerns about the interface's responsiveness. For example, they would have expected previews on rollover for any selection before actually validating it. On the opposite, experts understood the underlying computation cost and were tolerant regarding the non-instantaneous generation of views.

3.7 LIMITATIONS

3.7.1 *User interaction*

Our observational study revealed two aspects of the interface that might negatively impact the user experience. The first issue is related to the number of entities that are actually shown in a view. The automatic aggregation and selection of paths might give the impression that the view displays all entities in the collection. Although *S-Paths*' scoring strategy favours paths with high completeness, the semi-structured nature of data makes it very frequent to have irregularities in the data, and thus partial completeness only. While actual completeness is shown in the top-right corner of the interface, a participant mentioned that it should be made more salient. The second issue was raised by one of our lay users. While she liked the fact that *S-Paths* automatically selected views, she found it frustrating that the system did not remember the dimensions she had explicitly set when she was revisiting a given collection. One way of addressing this issue would be to take into account the navigation history in *S-Paths*' scoring strategy.

3.7.2 *Data processing*

We did set up *S-Paths* with seven datasets of varying size and with different characteristics: Nobel,[5] Data BnF,[6] ELI,[7] RISM,[8] John Peel Sessions,[9]

5 http://www.nobelprize.org/about/linked-data-examples/
6 http://api.bnf.fr/ (we ran our test on a 10 percent sample.)
7 http://data.public.lu/fr/datasets/legilux-journal-officiel-du-grand-duche-de-luxembourg/
8 https://old.datahub.io/dataset/rism
9 http://raimond.me.uk/resources/peel.tar.gz

Amsterdam Museum,[10] and Linked Movie DB,[11] revealing a few issues that *S-Paths*' approach can run into.

First, both ELI and RISM datasets lack `rdf:type` statements. As *S-Paths* relies on those statements to list candidate collections to start exploring from, we relied on the model to generate and store them in a small adjacent graph.

Second, *S-Paths* encounters performance issues with the BnF dataset. The high level of abstraction of the model, which makes the relevant paths potentially very long, combined with the very large number of entities, significantly increased the cost of each SPARQL query. This problem could be addressed by implementing mechanisms for query optimization. Furthermore, the granularity of the model resulted in a very large number of paths, some of them with very low completeness. To keep the matching algorithm reactive, we implemented an *ad hoc* solution that consisted of restricting the considered paths to the first fifty featuring the highest completeness at the start. A better solution would dynamically update the list of candidate paths whenever users focus on a new subset, as a given path's completeness can vary significantly from one subset to another. This would require implementing advanced graph exploration techniques to identify relevant paths without having to check them all after each new subselection.

3.7.3 *Available views*

Finally, in the John Peel Sessions dataset, *S-Paths* is unable to match paths with any view for several classes of resources. This is because values at the end of the paths consist of mostly-unique URIs or text values. In the current implementation, S-Paths does not provide aggregation mechanisms in such cases. Generic aggregation methods would make it possible to handle such situations. For example, text values could be grouped based on their first letters, while URIs could be grouped based on a common pattern. Such views would be rated low but would act as fallbacks when no other view is available.

3.8 DISCUSSION AND FUTURE WORK

Follow-your-nose style navigation on the Web of Data can be roughly compared to how users browse pages on the classic Web: they follow hyperlinks. This works well in open worlds of linked documents or linked data, because following a link is basically a question of dereferencing what is on the other side of that link. The process just gets repeated again and again, exploring

10 https://bitbucket.org/biktorrr/amlod/downloads/
11 https://old.datahub.io/dataset/linkedmdb

one path at a time. Things are not that simple when considering set-based navigation and *semantic paths*, as we do in *S-Paths*. We want to browse many resources simultaneously and look beyond the direct properties of those resources, following longer paths that lead to relevant information about them. This raises multiple practical questions. Which collections do we expose to bootstrap the exploration process? How deep down the paths do we go? Do we consider all possible paths? How many interlinked graphs do we traverse? Contrary to follow-your-nose navigation, considering (even theoretically) the open world in its entirety is not an option. Answering the previous questions necessarily entails delimiting *perimeters* in the data, that both the system and its users will be able to cope with.

S-Paths takes named graphs as the unit to delimit such perimeters. At first, this might seem contradictory with the very notion of a *Web* of Data, the whole point being to break out of data silos by having direct links between resources in different graphs from different datasets, and enabling the easy traversal of those links. Our purpose, however, is *not* to re-introduce artificial silos. It is rather to have some means to group collections meaningfully, and to identify coherent ensembles of *semantic paths* describing those resources. Small combinations of named graphs seem to be reasonable candidates to act as perimeters for such a purpose.

For now, *S-Paths* supports this approach by having the system connect to a single, local SPARQL endpoint that hosts one or more RDF graphs. For instance, the simple Nobel prize example used throughout the paper makes use of two linked graphs: the Nobel prize dataset itself, and an extract from DBpedia with the names and geo-coordinates of places, as well as pictures of people. While these two graphs are hosted behind the same SPARQL endpoint, we could imagine exploring graphs in distant endpoints using the SPARQL `SERVICE` clause.

However, this would come with significant performance issues, and would not be sufficient to enable users to navigate seamlessly across numerous linked graphs. Right now, graphs to include in the perimeter have to be declared manually. As we do not want to have all graphs merged into one huge perimeter (which would defeat the whole point of defining a perimeter and would be practically impossible anyway), we need *S-Paths* to be capable of dynamically redefining its perimeter in a way that is transparent to users: seamlessly adding new datasets to it based on the *semantic paths* followed, discarding datasets that are no longer relevant.

This also requires determining the best way to dynamically update the *semantic path* characterizations that *S-Paths* relies upon to rank and configure views. These characterisations need to be updated whenever the data change. But *S-Paths* cannot easily know when data hosted at remote endpoints change. Theoretically, the system is able to perform this analysis on the fly, since users can start browsing as soon as *S-Paths* has retrieved the first few paths, which only takes a few seconds. But having *S-Paths* permanently scan graphs is neither reasonable nor scalable. The characterisations also need to be updated when the perimeter changes, i. e., when graphs are added or removed. Knowing that a full analysis requires from several minutes to several hours depending on the number, size and structure of the graphs considered, an interesting hybrid approach would be to have data providers publish characterisations alongside their graphs (these are relatively small and simple JSON files). *S-Paths* would then only need to analyse connections between graphs when the perimeter changes. Achieving this would turn *S-Paths* into a full-fledged *linked data browser* combining set-based and follow-your-nose style navigation into a single tool.

Further evaluation will be needed. A key finding is that *S-Paths* lets users memorise the outlines of a dataset. It would be interesting to compare it with other RDF browsers in this respect. Another axis could be to try to understand which aspect of S-Paths supports memorising. This is an intricate question since it is a complex system, and we had to combine a number of strategies to make browsing collections even possible. Another promising result is that *S-Paths* can support users in finding ideas to reuse data. This is a crucial point for the adoption of RDF data, and as for memorising, it would be worth investigating which aspects of the tool makes it possible.

4

PATH OUTLINES

This chapter is a revised version of the paper co-authored with Olivier Corby, Jean-Daniel Fekete and Alain Giboin, published as a preprint at the time of writing [28]. My contribution included the initial idea, the design of the system, its implementation (except for LDPath API), its evaluation, and writing most of the paper. Olivier Corby implemented LDPath API. Alain Giboin and Jean-Daniel Fekete supervised the writing of the paper.

Path Outlines is a tool to support data producers in browsing path-based summaries of RDF datasets. Its interface is based on the broken (out)lines layout algorithm and the path browser visualisation. The tool is available as open source at gitlab.inria.fr/mdestand/spf and can be run online at spf.lri.fr.

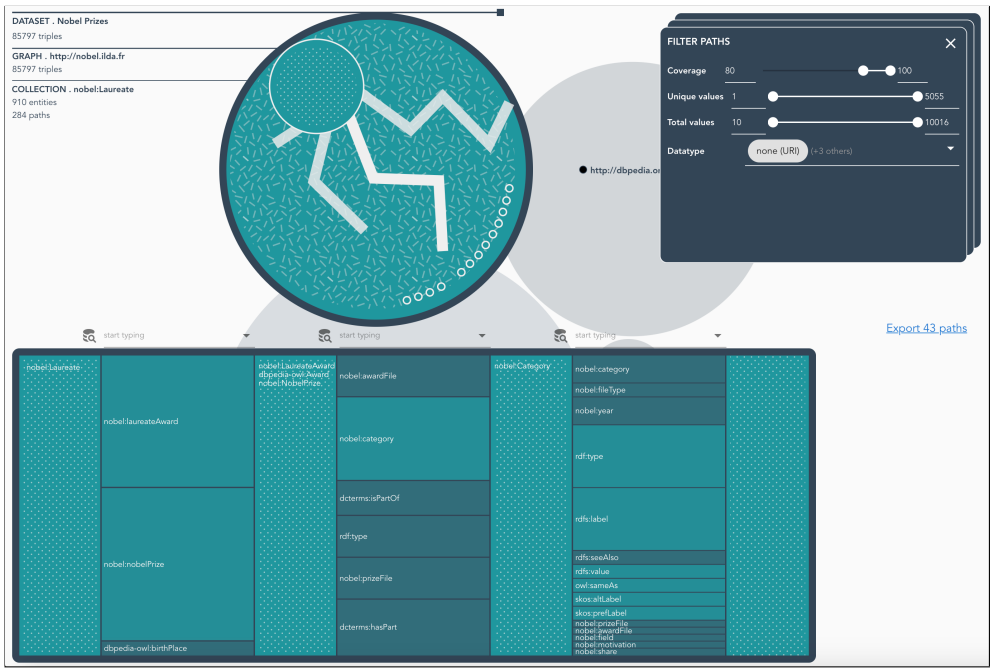


Figure 31: *Path Outlines* displays the analysis of paths of depth 3 for the `Laureates` collection in the Nobel dataset. The user has used the filter panel to see only the paths describing more than 80% of the entities: from the initial 80 paths of depth 3, only the 43 paths are left, other are filtered out. The user is currently hovering a property in the second column, which highlights in other columns all properties involved in sequences going through it. Clicking on this property would filter out properties that are not highlighted.

4.1 MOTIVATION

As she was navigating with *S-Paths*, the project manager of data.bnf.fr said that she would be curious to see the list of all paths. I realised that the summary provided by the paths could become the object of interest and that browsing the summary could support data producers in assessing the statements produced by their graph.

4.2 INTRODUCTION

RDF information is atomised in small units named *triples*. The triples can be combined to form complex statements depending on information needs. For instance, a triple in the Nobel Prizes dataset stating that "Marie Curie is affiliated to Sorbonne University", and another that "Sorbonne University is located in Paris" can be combined into "Marie Curie is affiliated to Sorbonne University in Paris". The chaining can be extended to other linked datasets. Such a structure is very expressive and powerful. A drawback of this expressivity, however, is that it makes it difficult for data producers to have an accurate overview of their data [121], and eventually improve it. *Our goal is to let data producers visualise the possible statements produced by a dataset, and browse them by meaningful chunks*.

In RDF, meaningful information about an entity is often 2 or 3 triples away from it, and current summary approaches fail to address chains of properties, also named *paths*. Most existing tools consider only triples, leaving aside all the statements that can be produced by chaining them. Other tools show summary graphs, presented as node-link diagrams, but their labels are difficult to read, and often laid out in various directions, barely allowing to follow paths. Given the large number of properties even in small databases, the node-link diagrams are either cluttered and unreadable or reduced to the most frequent classes and properties, offering a very partial overview of the paths available. They also provide metrics, but only at the triple level, displayed when users select an element in the diagram. In all cases, it is somehow possible to mentally recombine the paths, but this implies a high cognitive load [30]. Furthermore, combining statistics about triples does not provide statistics about paths. With a summary of the Nobel dataset stating that the database contains 911 laureates, 75% being affiliated to a university, and 525 Universities, 85% being located in a city, one could deduce that a laureate can have an affiliation that is located in a city, but there would be no way to know the percentage of laureates that actually do.

As Marchionini and Shneiderman stated in an early paper about hypertext systems, “key design issues include finding the correct information unit granularity for particular task domains and users” [70]. We posit that current RDF summary approaches are limited by their granularity, and that the *path* provides a meaningful granularity and expressivity to summarise Knowledge Graphs for data producers. We introduce *path outlines*, conceptual objects characterising sequences of triples with descriptive statistics. To provide an overview, and allow producers to determine which are of interest to them, we design and implement an interface supporting the Information Seeking Mantra: “Overview first, zoom and filter, then details-on-demand” [113]. Based on coordinated views with 2 novel visualisations, it allows to represent a very large number of *path outlines*, browse through them and inspect their metrics.

First, we introduce the difficulties to represent RDF data and visualise paths, and we discuss related work. Then we define the concept of *path outlines* to support path-based summaries, and we describe our API to analyse them. Next, we report on the interview of 11 data producers to evaluate their understanding and interest. After that, we present *Path Outlines*, our tool based on coordinated views representing the features of *path outlines*, to browse such summaries. Eventually, we conduct a use-case based evaluation of *Path Outlines* with 36 participants, in which we compare it with the Virtuoso SPARQL query editor as a baseline.

4.3 RELATED WORK

We discuss the difficulties to represent RDF data and paths, the types of summaries which are currently available, and the difficulty of writing and running queries for path-based summary information.

4.3.1 *Visualisation of paths in RDF data*

Node-link diagrams (Fig. 1-c) are often used to represent RDF datasets [97]. They accurately render their structure and are theoretically appropriate to easily recombine paths [84]. However, the readability of paths as sequences is very limited. Huang and Eades remark that people try to read paths from left to right and top to bottom, even when the layout and task require another direction [55]. Van Amelsvoort et al. demonstrate that the direction of elements influence reading behaviours [6]. Ware et al. show that good continuity, edge crossing and path length influence the effectiveness of visually following a path [130]. A specific type of node-link diagrams, node-link trees, seem to be more efficient for tasks related to following paths, traversing graphs [84,

85], and reading paths [68], probably because they constrain the flow in one direction. In their survey on the readability of hypertext, DeStefano and Lefevre mention several studies showing that the multiplication of possibilities impacts readability negatively [30], supporting the same idea. In contrast, PathFinder [89] lays flat all possible paths for the graph, allowing to read them easily, but this results in a very long list needing to be paginated even when the graph is small. For a data producer, gaining an overview of the paths in her own dataset is an unresolved problem. There is a need for a layout preserving their readability as sequences of statements to let users make sense of them, and enabling them to browse from the overview to the detail; this is what our tool does.

4.3.2 *RDF summaries for data curation*

An RDF summary is a concise description of the content of a dataset, sometimes characterised by descriptive statistics. We consider summaries that are meant for data producers, with the purpose of giving an overview of a dataset. Data profiling systems are tools presenting statistics about the data, such as LODStats [35], ProLOD [1, 20], LOUPE [76] or AETHER [80]. They typically present measures of atomic elements. While such summaries are complete and accurate, they give little information about the content. For a data producer, knowing that, for instance, 37% of all entities have a `rdfs:label` does not indicate what those labels are about, and is not very helpful to find missing information. Information becomes more meaningful with more context, like considering properties relatively to *subjects* with a specific `rdf:type` [61] (the number of `foaf:Document` having a `rdfs:label`), or to *objects* with a specific `rdf:type` [31, 32] (the number of `Persons` having a `birthplace` that is a `City`). This leads to more interpretable summaries, but is still limited to triples, not considering chains of statements.

Another approach consists in reconstituting a representative graph as the summary. Smallest representative graphs are used for machine consumption but are too big to be presented to users. User-oriented summaries limit summary graphs to the most represented classes, and the most represented direct properties between them [120, 121, 132], which make them graspable, yet very incomplete. Those summary graphs preserve access to chains of statements, but the statistics are produced and accessible at the triple level only, when users select an edge on the diagram. Furthermore, as we explained in the previous subsection, the readability of node-link diagrams is limited. Our approach considers an intermediate level as a unit for summaries:

the path. This allows us to summarise statements at a granularity that better matches data producers' needs, including the possible extensions of a path in interlinked datasets, and to provide metrics at this level of granularity.

4.3.3 *Querying summary information*

SPARQL, the main query language for RDF data, provides a syntax to query triples and paths in a graph. For instance, to query all property combinations composing the paths of depth 2 for entities of type `foaf:Document`, the query would look like `SELECT DISTINCT ?p1 ?p2 WHERE { ?s rdf:type foaf:Document. ?s ?p1 ?o. ?o ?p2 ?values }`. SPARQL also provides aggregation operators that can be applied to the elements we mentioned: entities, properties, triple patterns, possibly specifying the type of the subject and/or of the object, and deeper path patterns. However, queries combining aggregation and paths patterns are complex, and complex queries raise both technical and conceptual issues, as reported by Warren et al [131]. From a technical point of view, the cost of a query increases with the number of entities and the length of paths to evaluate. It is also impacted by the fact that a query is federated (targets several datasets), often resulting in network and server timeouts and errors that are difficult to manage in existing systems. From a conceptual point of view, summarising paths patterns for a large number of entities is not a simple mental operation. The task can be alleviated by tools to assist writing queries. YASGUI [102] offers autocompletion, syntax colouring and prefix handling. SPARKLIS [38] offers the possibility of discovering the model iteratively, enabling at each step to browse the available possibilities for extending the current path. However, such tools support only part of the task. They can be combined, which requires switching from one to another, and planning and thinking with them remains complicated and error-prone. As its name implies, almost everything in Linked Data is a link: entities, properties, classes and datatypes are URIs, which by definition are links [136]. However, technically, the connection between two datasets is made possible by joins: there is no explicit link between two datasets, but the fact that the same URI exists in both of them enables to query them jointly. Nonetheless, one tends to think of links, as illustrated by LOD Cloud[1] visualisation—frequently chosen to illustrate the Semantic Web—using a node-link diagram to represent datasets as nodes, and the presence of joins between two datasets as edges. This somehow ambiguous terminology adds difficulty when writing federated queries. Altogether, there

1 https://lod-cloud.net/

is a need for a tool to facilitate data curation by summarising and visualising paths in RDF data, including extensions to other datasets.

4.4 THE CONCEPT OF PATH OUTLINES

To provide metrics at the granularity needed by data producers to make sense of their datasets, we use *paths* and formalise the concept of *path outlines*, making them first-class citizens that can be described, searched, browsed, and inspected.

4.4.1 *Definition*

A *path outline* is a conceptual object providing descriptive statistics about a sequence of statements relative to a collection. It consists in: 1) a collection of entities sharing a similarity criterium (e.g., all the entities of class `Person`), for which at least one entity is the subject of a given sequence of properties, 2) the sequence of properties, 3) the set of objects at the end of this sequence, and 4) the set of measures relative to the collection and the objects, as schematised in Fig. 32.

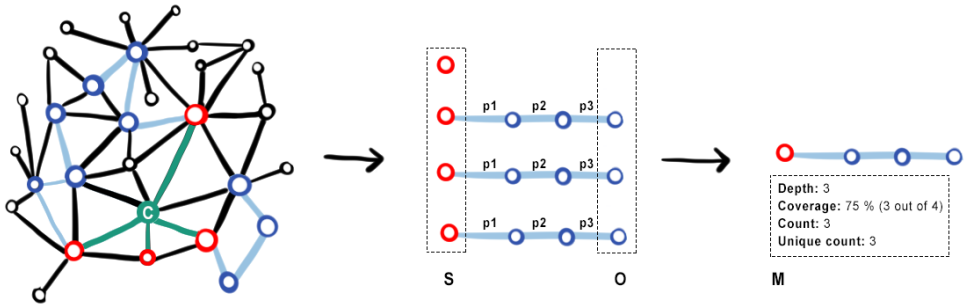


Figure 32: A *path outline*: for a collection S (red nodes) sharing a similarity criterium C (green node), a given sequence of properties $p_1/p_2/\ldots/p_n$ (light blue edges) leads to a set of objects O. S and O are characterised with a set of measures M. One can see that the starting entity for which the path is missing is taken into account in the summary.

A path of depth n is a sequence of n triples such that $(s, p_1, o_1), (o_1, p_2, o_2)$ $/\ldots/(o_{(n-1)}, p_n, o)$. Using the SPARQL property path syntax, this could be shortened as $(s, p_1/p_2/\ldots/p_n, o)$. To analyse a *path outline*, we start from a given collection S sharing a similarity criterion $c \in U$, and we consider a given sequence of properties, such that $\forall s \in S, (s$ `rdf:type` $c)\, \exists s \in S, \exists o \in O, (s, p_1/p_2/\ldots/p_n, o)$. O is the set of objects o at the end of the *path outline*. We compute a set of measures M relative to S and O, as described in Table 3. Each measure can be a literal value (e.g., a count), a distribution of values

Measure	Description
depth	number of statements between the set of entities S and the set of objects O
completeness	percentage of entities in the collection E for which this path exists
count	total number of objects in O
unique count	number of unique values or URIs for the objects in O
data types	only for literals: data type(s) of objects O at the end of the path
languages	only for string literals, if specified: list of languages of the objects O
min / max	for numerical values: minimum and maximum value for strings: first and last value, in alphabetical order

Table 3: Measures describing a *path outline*

(e.g., the number of unique values for URIs), or a range for numerical values.

To write a path, we defined a syntax inspired from XPath [23] (Fig. 33). The template string is similar to an XPath query selector: it is a pointer to designate the chains of triples corresponding to the query and summarised by a *path outline*. The syntax is easy to parse at a glance: the elements are separated by a slash (reminiscent of the syntax of file paths in operating systems). The first chunk is the similarity criterium, the number of stars indicates the depth of the path, and they create a visual articulation to separate the other chunks, corresponding to the properties forming the path. It already forms a graphical object revealing the articulations that will show in our visualisations.

For instance, considering the full Nobel dataset, from which a sample is presented in Fig. 1, a *path outline* of depth 1 relative to the set of laureates, and describing those whose birth date is known in the dataset, can be expressed as `nobel:Laureate/foaf:birthday/*/`. Its completeness is 96% [2] and could be expected to be 100% after data curation since the information is likely to be available in external sources. Fig. 33 shows the *path outline* of depth 3 describing the laureates having an affiliation, which has the city, which has a similarity link to another resource. In this case, the completeness rate is unlikely to reach 100%, as some laureates might not have an affiliation. The number of unique values is higher than the number of laureates having an affiliation, some of them having multiple affiliations.

2 http://data.nobelprize.org/sparql accessed on 01/03/2020

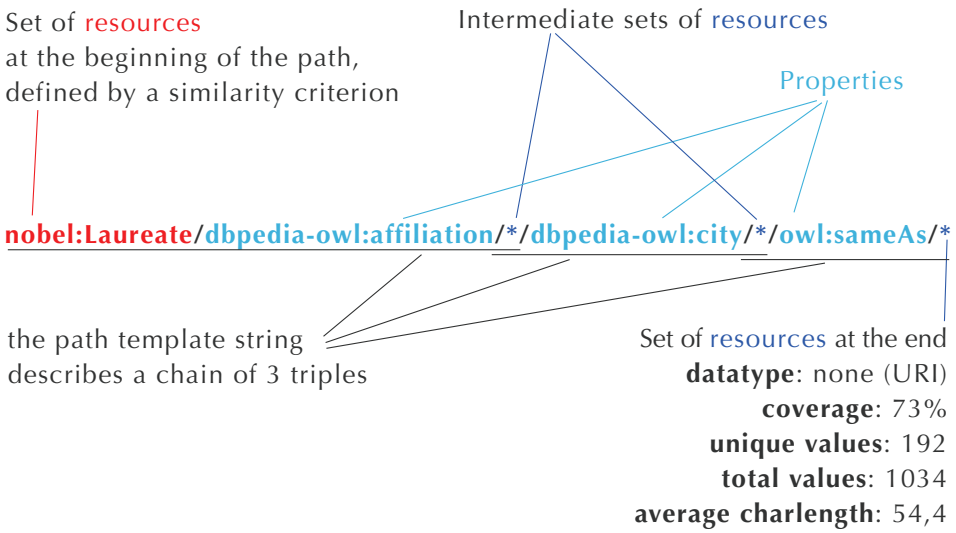


Figure 33: Template string for a *path outline*, summarising the Nobel laureates having an *affiliation*, *located in a city*, *having a similarity link* to another resource. Intermediate sets of resources are designated by stars, indicating that they can be of any type. Those resources are both the objects of the preceding predicate, and the subjects of the next.

4.4.2 *LDPath API*

To analyse the paths, we developed a specific extension to a semantic framework for Knowledge Graph querying[3]. Given an input query, it discovers and navigates paths in a SPARQL endpoint by completing the input query with predicates that exist in the endpoint. LDPath first computes the list of possible predicates and then, for each predicate, counts the number of paths. This is done recursively for each predicate until a maximum path length is reached. The values at the end of each path are analysed to retrieve the features listed in Table 3. LDPath can also, for each path, count the number of joins of this path in another endpoint, and compute the list of possible predicates to extend the path by one statement. The values at the end of the extension are also analysed. The software package recursively rewrites and executes SPARQL queries with appropriate service clauses. The API of this extension is made available for other purposes and can be queried independently of *Path Outlines*[4].

4.5 USER STUDY 1: VALIDATING THE APPROACH

One of the authors has several years of experience in the Knowledge Graphs community. Taking inspiration from her experience and situations she had ob-

3 reference to a paper, anonymised for submission
4 link to the API, anonymised for submission

served in professional semantic web meetups and conferences, we designed 6 real-world task scenario involving finding or browsing path-based metrics. For instance, the second scenario was: “Find the datatypes of the set of values at the end of a path. For example, identify if at the end of certain properties there are alternatively dates or URIs, or check if the date formats typed as such and valid”. We interviewed 11 data producers to validate our approach.

4.5.1 *Participants*

We conducted a fifteen to thirty-minute interview with 11 RDF data producers recruited via email calls on Semantic Web mailing lists and Twitter. Participants belonged to industry (4), academia (4) and public institutions (3). The datasets they usually manipulated contained data from various domains, ranging from biological pathways to cultural heritage through household appliances. All participation was voluntary and without compensation.

4.5.2 *Set up and procedure*

The interview was supervised online through a videoconference system. We presented each task scenario. We asked participants if they did already perform similar tasks; and if so, how often and by which means; if not, for what reason. We asked them if they would be interested in a tool supporting such tasks. Eventually, we asked them if they could think of other similar or related tasks that would be useful for them.

4.5.3 *Results*

We collected answers in a spreadsheet and analysed them with R.

4.5.3.1 *Current usage of path-based metrics*

A few participants already performed tasks that were similar to the ones in our scenarios, as reported in Fig. 34. They used SPARQL query editors (16)[5] or *content negotiation* in the browser (3): they pasted a URI in the browser to see the triples describing it, and copy-pasted other URIs to continue the chaining, entity by entity. The main reason given for not performing a task or performing it too rarely was *no tool* (14). Those tasks are actually possible with SPARQL, but participants either did not know how to write the queries or regarded it as so complicated that they would not even consider it as an option. The second

5 the counts in this paragraph correspond to the number of scenarios, not to the number of participants

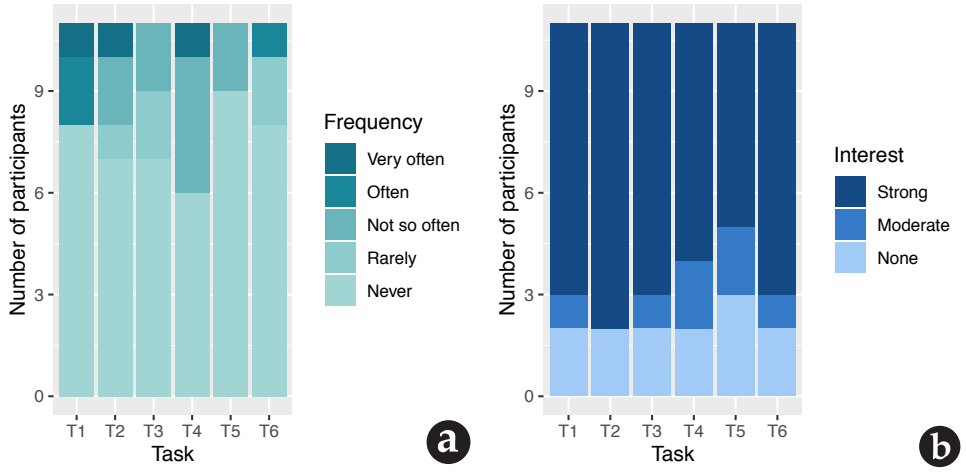


Figure 34: Usage and interest of data producers regarding the scenarios: a) they hardly ever perform similar tasks, b) but would be very interested in a tool supporting them.

main reason was time concerns (13): the task was regarded as doable, but it would have taken too long to write such queries.

4.5.3.2 *Interest for path-based summaries*

Two participants had difficulties in relating to the scenarios. Their use of RDF data was focused on querying single entities rather than sets. They did not feel the need for an overview (although one changed his mind, as explained in Sect. 4.7.6.4). Most other participants declared a strong interest (Fig. 34): 3 had already well identified their needs, and the others sounded really enthusiastic that we were able to formulate them. Six participants spontaneously mentioned clearly seeing the interest of a tool supporting similar tasks for data reusers, in a discovery context. Only one participant suggested a related task: *identify outliers in values of paths typed as numerical values*, involving more advanced metrics on paths than the one we had mentioned.

This interview confirmed the interest of data producers for path-based summaries, and the fact that for those who were already gathering very similar information, a SPARQL query editor was the baseline.

4.6 PATH OUTLINES, THE TOOL

To let users browse *path outlines*, we designed an interface based on coordinated views with two new visualisations (Fig. 31). We present the design requirements and the design rationales for the interface, followed by 2 scenarios of use.

4.6.1 *Design requirements: from overview to detail*

The process of browsing through an information space can be well described by the Information Seeking Mantra: “Overview first, zoom and filter, then details-on-demand” [113]: The tasks involved in this navigation paradigm are: ‘find the number of items’, ‘see items having certain attributes’, and ‘see an item with all its attributes’. The overview is meant to provide context to users, to ‘gain an overview of the entire collection’. There are several levels of contexts for paths: the dataset, and the starting set of entities. It shall also give them an idea of the main features of the items in the collection, which will allow them to determine what is of interest and what is not, and to progress through the collection, ‘zoom in on items of interest and filter out uninteresting items’, and finally ‘select an item or group and get details when needed’. The particular difficulty with *path outlines* is that their features are both metrics related to them and the sequences of properties composing them. To address this specificity, our interface combines several coordinated views.

4.6.2 *Interface: coordinated views to display complex objects*

The interface relies on two new visualisations: the *broken (out)lines* algorithm – extending a circle packing layout, and the *path browser*. They are coordinated with several filter panels.

4.6.2.1 *Context overview: circle packing and broken (out)lines*

As users open *Path Outlines*, they see several datasets laid out with a circle packing algorithm [25]. Their size is mapped to the number of triples they contain (Fig. 37-1). Using the filter panel (Fig. 37-2), they can select a specific size range or search by name. When they open it in the foreground (Fig. 38-3), datasets that are linked to it also come to the foreground, as small bullets laid out on the side (Fig. 37-8). The different collections sharing the same `rdf:type` in the main dataset are laid out inside in another circle packing, their size corresponding to the number of entities (Fig. 38-4). The filter panel allows to filter by size and name (Fig. 38-6). As they click on one to open it, other collections become smaller and are aligned on the side to be easily available (Fig. 39-8). The available *path outlines* depths (Fig. 39-7) are laid out with the broken (out)lines algorithm. It relies on simple geometrical principles. The algorithm is described in Fig. 35. It is inspired by systems that present an overview of a graph with different possible *cuts* in it, that can be inspected in a coordinated view [2, 8]. The shape of *broken (out)lines* is reminiscent of a

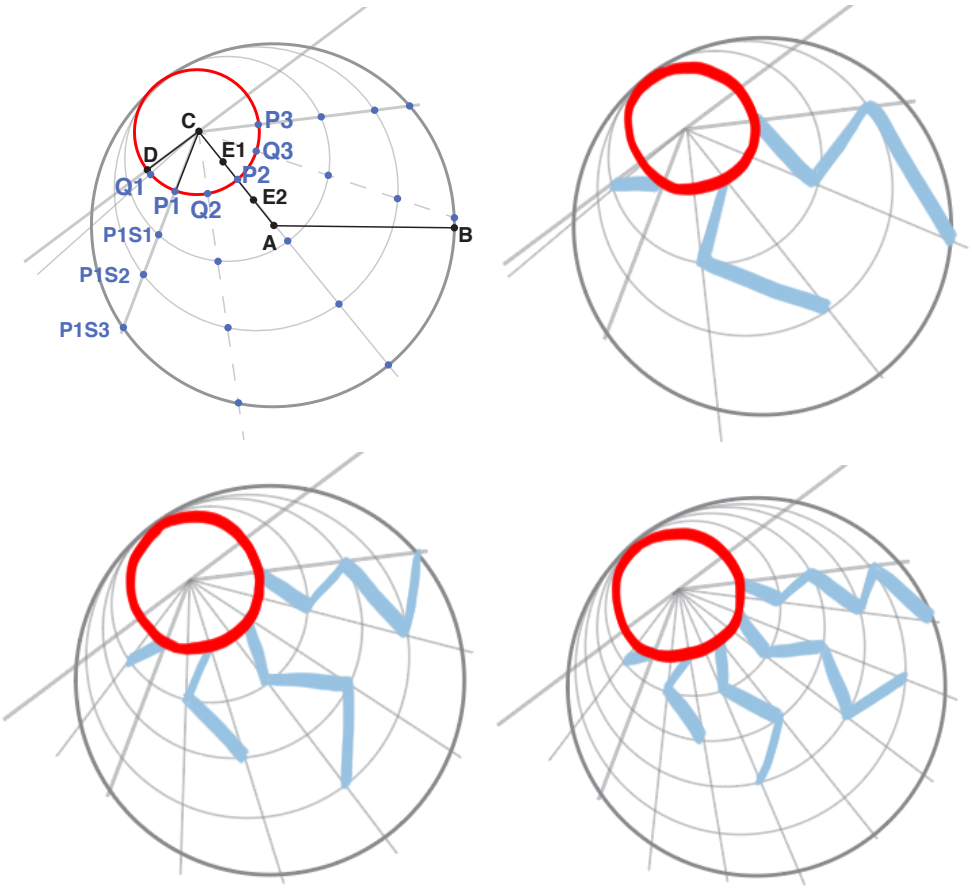


Figure 35: Broken (out)lines algorithm: broken (out)lines are drawn and positioned according to the maximum depth of *path outline*, using geometrical principles to fit in the circle.

Algorithmus 1 : Pseudo-code to draw the broken lines

```
// Initialise the constants
A.x       // pos x of the main circle
A.y       // pos y of the main circle
A.r       // radius of the main circle
BAC       // angle in degrees, to position
          // the (small red) entities circle
maxdepth  // number of broken outlines
DCP_1     // angle to reduce the scope
// Compute radius and position of entities circle
C.r = A.r/3;
C.x = A.x + A.r + (A.r × Math.cos(BAC));
C.y = A.y + A.r + (A.r × Math.sin(BAC));
for n = 1 to maxdepth do
    // Position P_n points on the entities circle
    // 1. Compute angle
    P_1CP_2 = (180 − (DCP_1 × 2))/(n − 1);
    // 2. Compute position of P_npoints
    P_n.x =
     C.x + (C.r × Math.cos((BAC + 90 + DCP_1) + P_1CP_2 × (n − 1)));
    P_n.y = o2.y + (o2.r × Math.sin((BAC + 90 + DCP_1) + P_1CP_2 ×
     (n − 1)));
    Q_n.x = o2.x + (o2.r × Math.cos((BAC + 90 + DCP_1) + P_1CP_2 ×
     (n − 2) + 1/2P_1CP_2));
    Q_n.y = o2.y + (o2.r × Math.sin((BAC + 90 + DCP_1) + P_1CP_2 ×
     (n − 2) + 1/2P_1CP_2));
    // Compute radius and position
    // of grey circles E_n
    E_n.r = C.r + n × ((A.r − C.r)/(n + 1));
    E_n.x = A.x + ((A.r − E_n.r) × Math.sin(BAC));
    E_n.y = A.y + ((A.r − E_n.r) × Math.sin(BAC));
    // call function to find intersections
    // between CP and CQ lines and E circles
    for m = 1 to maxdepth do
        P_nS_m = findIntersection(line CP_n, circle E_m);
        Q_nS_m = findIntersection(line CQ_n, circle E_m);
    end
end
```

node-link diagram so that users can relate it to a representation they already know, and understand what is displayed below in the *path browser* (Fig. 39-9). Associated with the circle, they form a glyph [39], a simple symbol meant to be readable, and yet encoding important attributes of the data. By default, *path outlines* of depth 1 are selected.

4.6.2.2 *Zoom and filter: the* path browser *and filter panel*

Path outlines being composed of sequences of properties, it would be possible to represent them with a Sankey diagram [101, 107], as shown in Fig. 36-a. However, the number of *path outlines* that could be displayed would be limited, and it would be difficult to follow the edge that the labels relate to and to identify sequences. The *path browser* keeps the links, but merges the nodes so that the links do not need to be curved any more: they become rectangles (Fig. 36-b). Merged nodes are turned into vertical rectangles representing entities, allowing to display their `rdf:type` when it is known. The vertical rectangles are aggregated by property, and the height of a rectangle is proportional to the frequency of the property in all paths. This allows prioritising the readability of the best-represented properties. Even in extreme cases, where the number of properties is very high, the coordination with the filter panel (Fig. 39-10) allows to reach a readable state very quickly: users are typically interested either in inspecting paths that are shared by most of the entities, to know which data can be queried or in finding entities that are not well shared, in order to fix them. Users can also use the panel to filter on other features and gain an overview of the available range for each feature.

The information about sequences is made available through interactivity: hovering a property highlights all possible sequences going through it (Fig. 31); clicking on it selects this property, and filters out properties which are not highlighted. Selected properties form a pattern, and all *path outlines* that do not match this pattern are filtered out. Users can also form the pattern using the search fields with autocompletion above each column. Furthermore, patterns and statistical filters can be combined.

4.6.2.3 *Details-on-demand: the detail panel*

When users hover or select a single *path outline*, its statistical description appears in the statistical panel (Fig. 39-11). This panel also offers a list of linked datasets to which the selected *path outline* can be extended. When a linked dataset is selected, a column is added on the right (Fig. 40-12), to let them browse possible extensions to the *path outline*. The filter panel (Fig. 40-

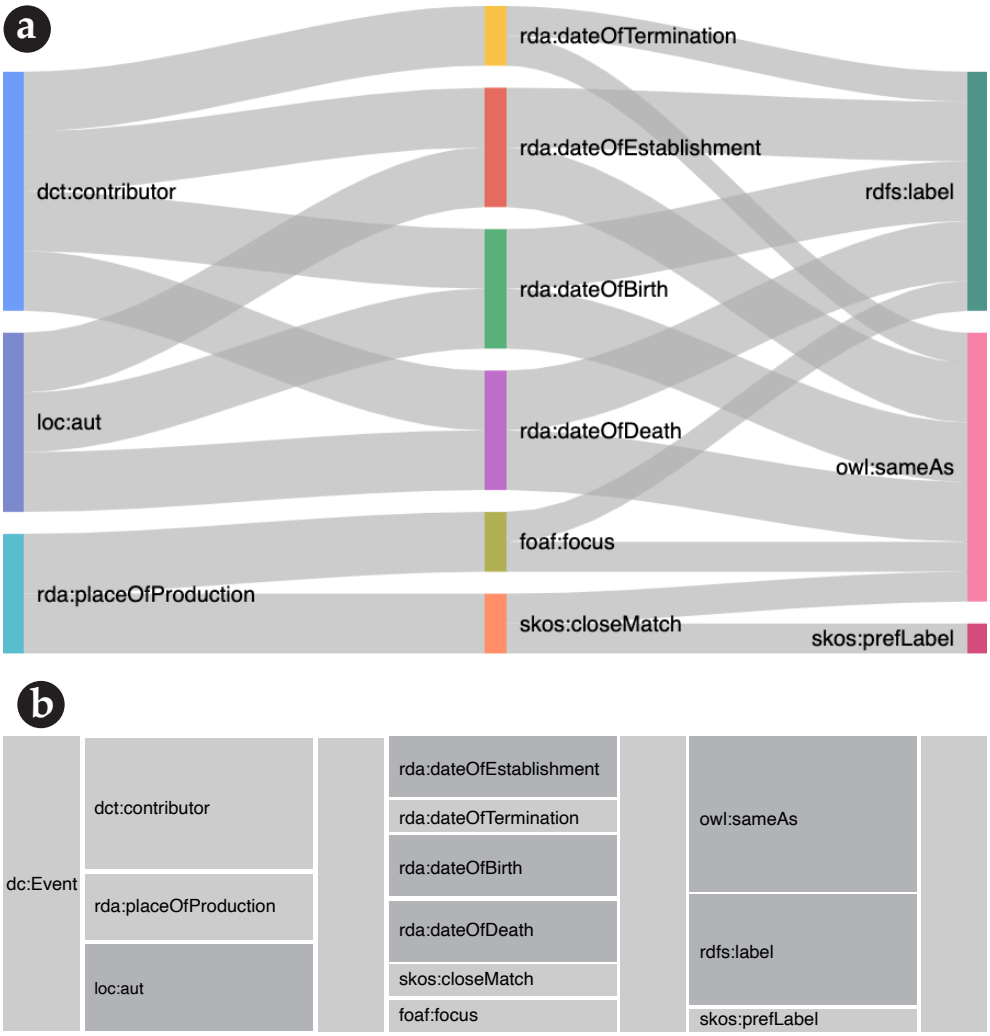


Figure 36: The same 18 *path outlines* displayed in a Sankey Diagram (a) and the *path browser* (b). Hovering the property `loc:aut` highlights all matching sequences.

13) and statistical panel (Fig. 40-14) now apply to the extended *path outlines*. A line shows the target dataset, inviting users to click it and explore its *path outlines*.

4.6.3 *Scenario of use*

4.6.3.1 *Scenario 1*

A member of the DBpedia community would like to check the quality of the data describing music albums in the DBpedia dataset. She opens *Path Outlines*, searches `DBpedia` in the filter panel (Fig. 37-2). A dozen of datasets remain, all other are filtered out (Fig. 37-1). Hovering them, she can see each one corresponds to a different language. She clicks on the French version, which opens in the foreground (Fig. 37-3). To find music albums among the

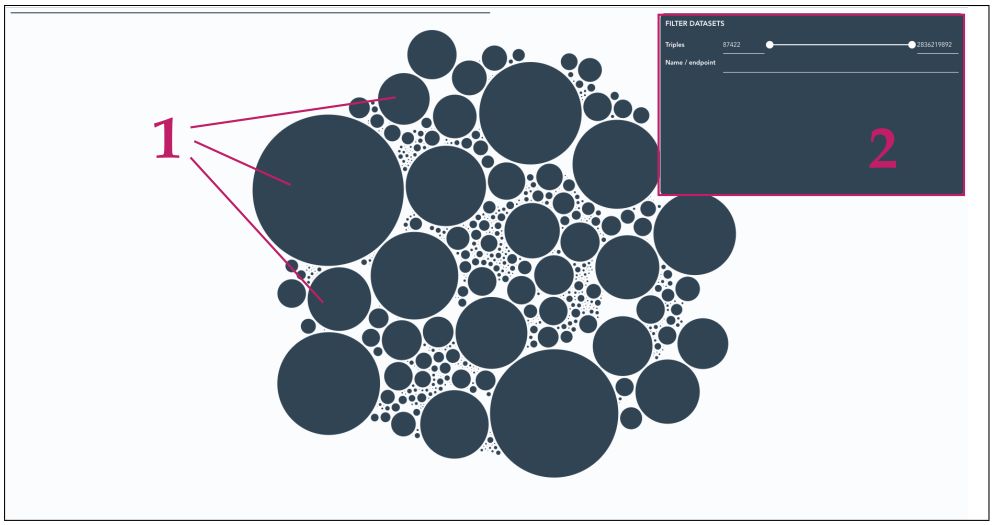


Figure 37: From overview to detail. At launch, the tool presents all available datasets (1), users can filter them by size and name (2).

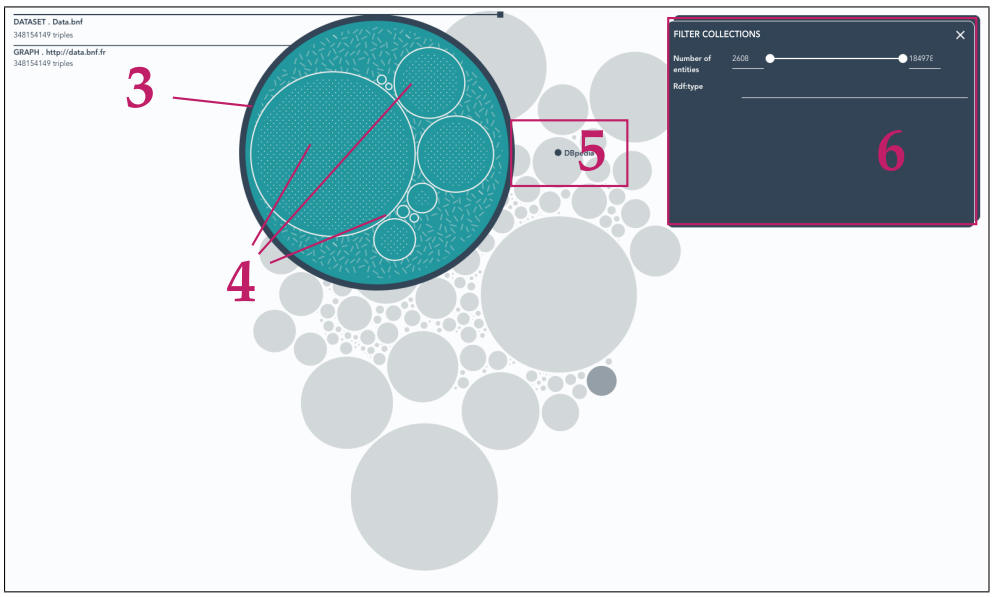


Figure 38: When a dataset is selected, interlinked datasets are placed aside (5), and collections (4) are presented inside the open dataset (3). Users can filter collections by size and name (6).

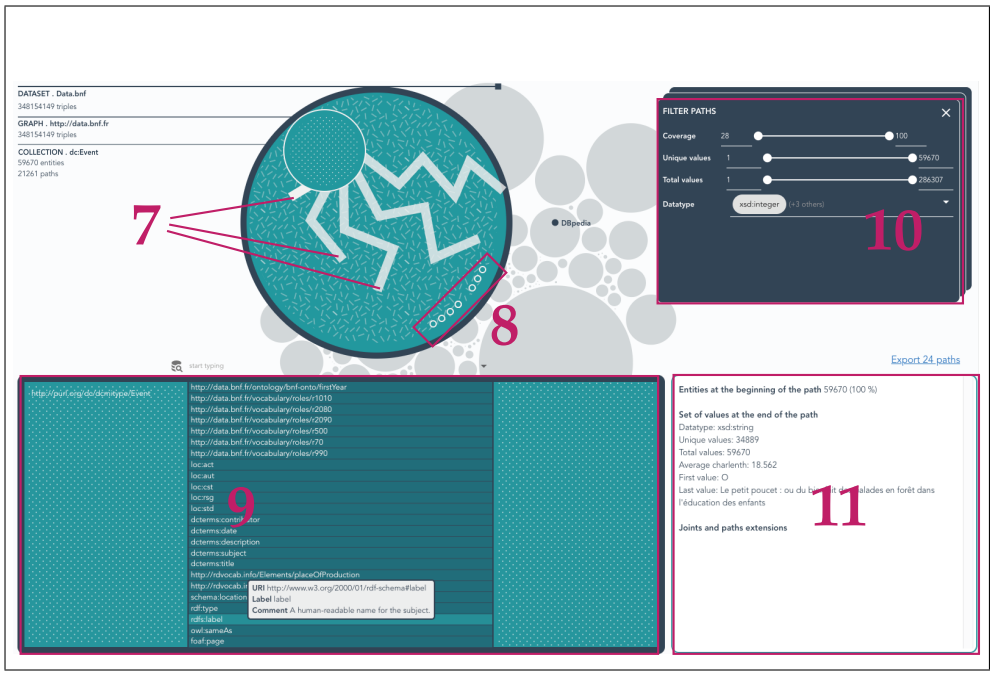


Figure 39: When a set is selected, *path outlines* of depth 1 are displayed in the *Path Browser* (9), and users can select other depths (7). Users can filter paths by statistical feature or name (10). When a single path is hovered or selected, details are available in the detail panel (11).

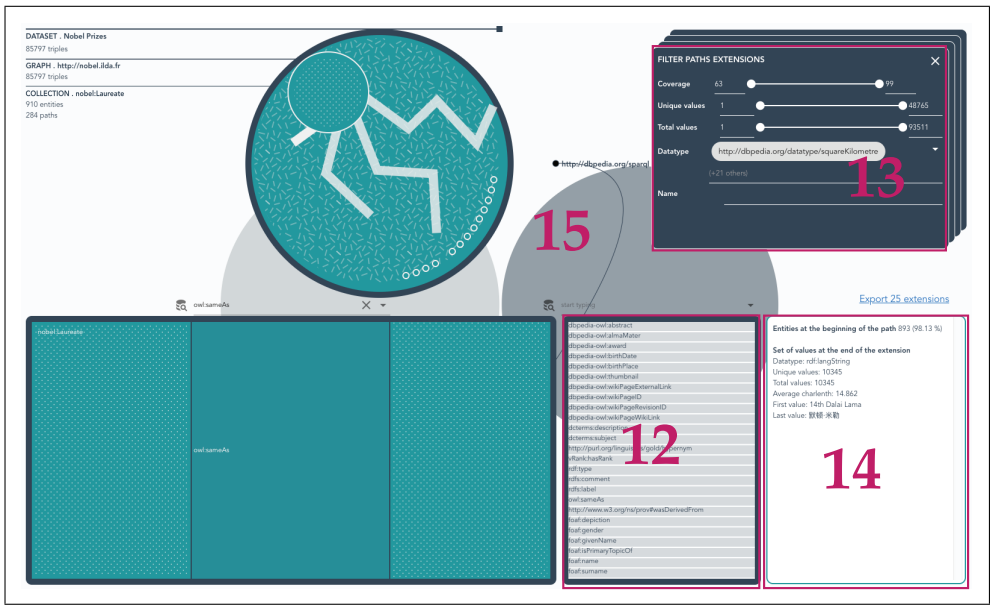


Figure 40: When an external dataset is selected, extensions of the current path in this other dataset are presented (12).

many collections, she types `music` in the filter panel (Fig. 37-6). Five collections correspond to this keyword (Fig. 37-5), she hovers them and identifies `schema:MusicAlbum`, which she selects. This isolates the set, displays its broken (out)lines (Fig. 37-7), and opens the path browser (Fig. 37-8). By default, paths of depth 1 (such as `http://dbpedia.org/ontology/composer` or `http://dbpedia.org/ontology/format`) are displayed. The interface announces that there are more than 41 000 albums, with 87 paths of depth 1. She wants to check properties with bad completeness, to see if there is a reason for this. She uses the cursor in the filter panel (Fig. 37-10) to select paths with completeness rate lower than 10%. She hovers available paths and inspects their completeness. She notices that the property `http://fr.dbpedia.org/property/writer` is used only once. A property which sounds very similar, `http://dbpedia.org/property/writer`, is used more than 800 times. To identify the entity she needs to modify, she clicks on the button “See query” that opens the SPARQL endpoint in a new window, prefilled with a query to access the set of DISTINCT values at the end of the path. She will now do similar checks with other paths of depth 1 and paths of depth 2.

4.6.3.2 *Scenario 2*

A person in charge of the Nobel dataset would like to know what kind of geographical information is available for the `nobel:Laureates`. Could she draw maps of their birthplaces or affiliations? She knows there are no geo-coordinates in the dataset, but some should be available through similarity links. She opens *Path Outlines*, searches `nobel` in the filter panel, and opens her dataset. She then selects the `nobel:Laureates` start set. She starts to look for laureates having an affiliation aligned with another dataset. She selects paths of depth 3. In the first column, she types `affiliation`. This removes other properties than `nobel:affiliation` from this column, and properties which are not used in a path starting with `nobel:affiliation` from other columns. Among properties remaining in the second column, she can easily identify `dbpedia:city`, which she selects. In the third column, she selects `owl:sameAs` property. A single path is now selected, summary information appears in the inspector: 72% of the laureates have an affiliation aligned with an external dataset. She selects the link to display extensions in DBpedia. A list of 78 available properties to extend the path in DBpedia appear. She types *geo* in the search field. A list of 4 properties containing *geo:lat* and *geo:long* remains. She inspects the summary information of the extended paths: only 32% of the laureates have geo-coordinates in DBpedia. She re-

peats the same operations for birthplaces: 96% have a similarity link to an external dataset, among which 61% have geo-coordinates in DBpedia. She can now assess the completeness of the dataset regarding the laureates and their locations, and plan to fix the missing information.

4.6.4 *Implementation*

The front-end interface is developed with NodeJS, it uses Vue.js and d3.js frameworks.

4.7 USER STUDY 2: EVALUATING PATH OUTLINES

We designed an experiment to compare *Path Outlines* with the virtuoso SPARQL query editor (hereafter SPARQL-V). Although comparing a non-graphical tool with a graphical tool can be controversial, it is the relevant baseline in this case: a SPARQL editor is the only way to fully perform the tasks we are evaluating as of today, and this specific editor is the most used by our target users, as confirmed by participants in study 1. The experiment was a $2 \times 2 \times 3$ within-subject controlled experiment, with a mixed design (counterbalanced for the two first variables, and ordered for the last one), to compare *Path Outlines* with SPARQL-V. The first independent variable was the tool, with two modalities: Path Outlines vs SPARQL-V. The second independent variable was the dataset, with two modalities: Nobel dataset vs Persée dataset. The third independent variable was the task, with 3 modalities: 3 tasks ordered by difficulty (with small adaptations to the dataset). The dependent variables we collected were the perceived comfort and easiness, the execution time, the rate of success and number of errors, and the accuracy of memorising the main features of a dataset. Our hypotheses were:

H1: *Path Outlines* is easier and more comfortable to use than SPARQL-V

H2: *Path Outlines* leads to shorter execution time than SPARQL-V

H3: *Path Outlines* leads to better task completion and fewer errors than SPARQL-V

H4: *Path Outlines* facilitates recalling the main features of a dataset compared to SPARQL-V

4.7.1 *Participants*

We recruited 36 participants (30 men and 6 women) via calls on semantic web mailing lists and Twitter, with the requirement that they should be able

to write SPARQL queries. 5 participants in the interview also registered for the experiment. Job categories included 12 researchers, 10 PhD students, 9 engineers and 3 librarians. 29 produced RDF data and 31 reused them. Their experience with SPARQL ranged from 6 months to 15 years, the average being 5.07 years and the median 4 years [6]. 12 rated their level of comfort with SPARQL as *very comfortable*, 11 as *rather comfortable*, 10 as *fine*, and 3 as *rather uncomfortable*. 18 used it *several times a week*, 13 *several times a month*, 2 *several times a year* and 3 *once a year or less*. 23 of them listed Virtuoso among the tools they were regularly using. All participation was voluntary and without compensation.

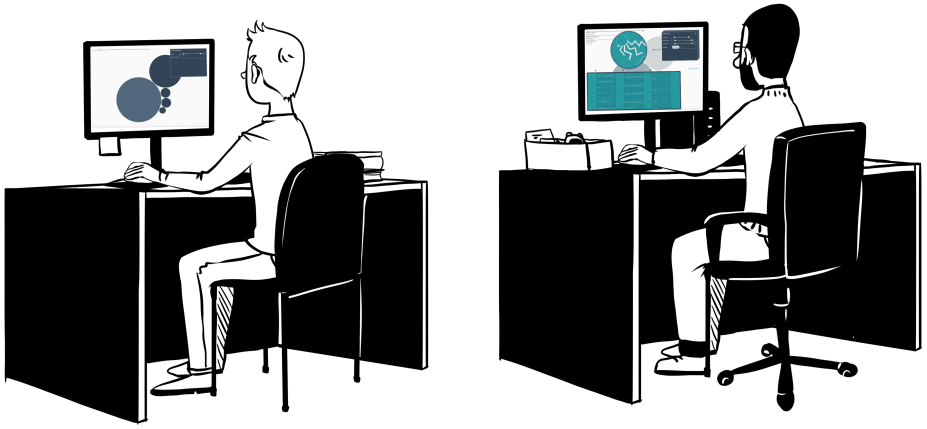


Figure 41: Participants to our evaluation, corresponding to expert persona: *data producers* and *data reusers*.

4.7.2 *Setup*

The experiment was mostly supervised online through a videoconferencing system. It was run face-to-face for 3 participants. We used an online form to guide participants through the tasks and collect the results. The form provided links to our tool, to a web interface developed in JavaScript, and to a SPARQL endpoint we had set up for the experiment. In 5 cases, due to restrictions in the network, we replaced the endpoint by the Nobel public endpoint. We used two datasets, Nobel and Persée, which had been analysed with our tool and are hosted in our endpoint. 2 participants stopped after 2 tasks because of personal planning reasons, so we asked the last two participants to complete only 2 tasks to keep the 4 configurations balanced for all tasks.

6 SPARQL has existed since 2004, the standard was released in 2008

4.7.3 *Tasks*

We designed 3 real-world tasks, ordered by difficulty. They involved the 3 nuclear tasks that our interface supports, combined in different ways. On Nobel Dataset, Task 1 (T1) was: Consider all the awards in the dataset. For what percentage of them can you find the label of the birthplace of the laureate of an award? Task 2 (T2) was: Consider all the laureates in the dataset. Find all the paths of depth 1 or 2 starting from them and leading to a piece of temporal information. Indicate the data type of the values at the end of the path. Task 3 (T3) was: Imagine you want to plot a map of the universities. The most precise geographical information about the universities in the dataset seems to be the cities, which are aligned to DBpedia through similarity links `owl:sameAs`. Find one or several properties in DBpedia (`http://dbpedia.org/sparql`) that could help you place the cities on a map. The tasks on Persée Dataset were equivalent, with small adaptations to the context.

4.7.4 *Procedure*

We sent an email to the participants with a link to the video conference. As they connected, we gave them a link to the form. They were invited to read the consent form. We started with a set of questions about their experience with SPARQL. Then we introduced the experiment and explained how it would unfold. The first task T1 was displayed, associated with a technique and a dataset. We read it aloud and rephrased the statement until it made sense to the participants. Participants were asked to describe their plan before they performed the task. We rated the precision: 0 for no or very imprecise planning, 1 for imprecise planning, 2 for very precise planning. The time to perform the task was limited to eight minutes. If they were not able to complete in time, they were asked to estimate how much time they think they would have needed. Then they rated the difficulty of the task and the comfort of the technique. The next task was the equivalent task T1 associated with the other technique on the other dataset. We counterbalanced the order of the technique and dataset factors, resulting in 4 configurations. After the set of two equivalent tasks, participants were asked which environment they would choose if they had both at their disposal for such a task. The same was repeated for tasks T2, and then T3. In the end, participants answered a multiple-choice query form about the general structure of a dataset: number of triples, classes, paths of depth 1 and 2. To finish with, they were invited to comment on the tool and make suggestions.

4.7.5 *Data collection and analysis*

We collected the answers to the form, and analysed with R.

4.7.5.1 *Perceived comfort and easiness*

In general, participants found *Path Outlines* more comfortable than SPARQL-V (Fig. 43a). Several participants said that they would need more time to become fully comfortable with *Path Outlines*. Five minutes of practice was indeed a very short time, but the level of comfort reported with *Path Outlines* is already quite satisfactory. The level of comfort reported when performing tasks with SPARQL-V was lower than the level initially expressed. We interpret this as being due partly to the fact that it is uncomfortable to code when an experimenter is watching, and partly to the difficulty of the tasks. Being very familiar with SPARQL does not mean being familiar with queries involving both sets of entities and deep paths. This supports the idea that a specific tool for such tasks can be useful even for experts. Three users mentioned being less comfortable with Virtuoso than with their usual environment. However, Virtuoso was the tool most frequently listed as usual by participants (23). Participants perceived the same tasks as being easier when performed with *Path Outlines* than with SPARQL-V, as shown in Fig. 43b. We think this is because *Path Outlines* enables them to manipulate the paths directly, saving them the mental process of reconstructing the paths by chaining statements and associating summary information to them. A participant wrote us an email after the experiment to thank us for the work, saying that “such tools are needed due to the conceptual difficulties in understanding large complex datasets”. Those results are in agreement with H1.

4.7.5.2 *Task execution time*

We counted 8 minutes for each timeout or dropout. Participants were quicker with *Path Outlines* on the three tasks, as shown in Fig. 43e, in agreement with H2. We applied paired sample t-tests to compare execution time, with a log transformation to normalize the distribution, with each technique for each task. There was a significant difference in the three tasks: T1: $t = 14.368, p = 3.026^{-16}$, T2: $t = 6.3173, p = 2.956^{-7}$, T3: $t = 17.467, p < 2.2^{-16}$, which shows that participants were significantly faster on each task with *Path Outlines* than with SPARQL-V. The effect size is very large: the median is 480s with SPARQL-V vs. 119s with *Path Outlines* on T1, 472.5s with SPARQL-V vs. 215s with *Path Outlines* on T2, and 480s with SPARQL-V vs. 146.5s with *Path*

Outlines on T3. We asked those who did not complete the tasks to give an estimation of the additional time they would have needed. We did not use self-estimations to make a time comparison since not all participants were able to answer, and such estimations are likely to be unreliable since time perception and self-perception are influenced by many factors. However, we report them as an indicator: for participants with a very precise plan, it ranged from 30 seconds to one hour; with an imprecise plan, it ranged from 15 seconds to 45 minutes; and with no plan, it ranged from 4 minutes to several hours. Task 2 required them to look at paths of two different depths. Although participants were longer on this task, *Path Outlines* still outperformed Virtuoso SPARQL query editor, but several participants expressed the wish to see both depths at the same time.

4.7.6 *Results*

4.7.6.1 *Task completion and errors*

Using our tool, only one participant timed out on task 2; all others managed to complete each of the tasks within 8 minutes. With SPARQL-V, there were 37 dropouts (9 on T1, 10 on T2 and 18 on T3) and 15 timeouts (9 on T1, 5 on T2 and 1 on T3). Among the tasks completed in time, 28 did had erroneous or incomplete results with SPARQL-V (11 on T1, 13 on T2 and 5 on T3) versus 13 with our tool (on T2), as summed up in Fig. 43c. The main errors on T1 were that some participants counted the number of paths matching the pattern instead of the number of documents having such paths (either by counting values at the end of the paths or by counting entities without the DISTINCT keyword). It occurred 9 times in SPARQL-V, and never with our tool. Four participants were close to making the same mistake but corrected themselves with SPARQL-V, and one did so with our tool. Another error occurred only once with SPARQL-V: the participant started from the wrong class of resource. T2 presented the particular difficulty that temporal information in RDF datasets can be typed with various data types, including `xsd:string` and `xsd:integer`. The most common error was to give only part of the results, either because of relying on only one data type, or because it was difficult to sort out the right ones when displaying all of them. It occurred 12 times with both techniques. The mean percentage of correct results was 75% with our tool, versus 50% with SPARQL-V. With SPARQL-V, one participant happened to give all paths as an answer, including non-temporal ones, which we regarded as a partial success. For T3, one participant gave an answer that did not meet the requirement with SPARQL-V, stating that it would be too

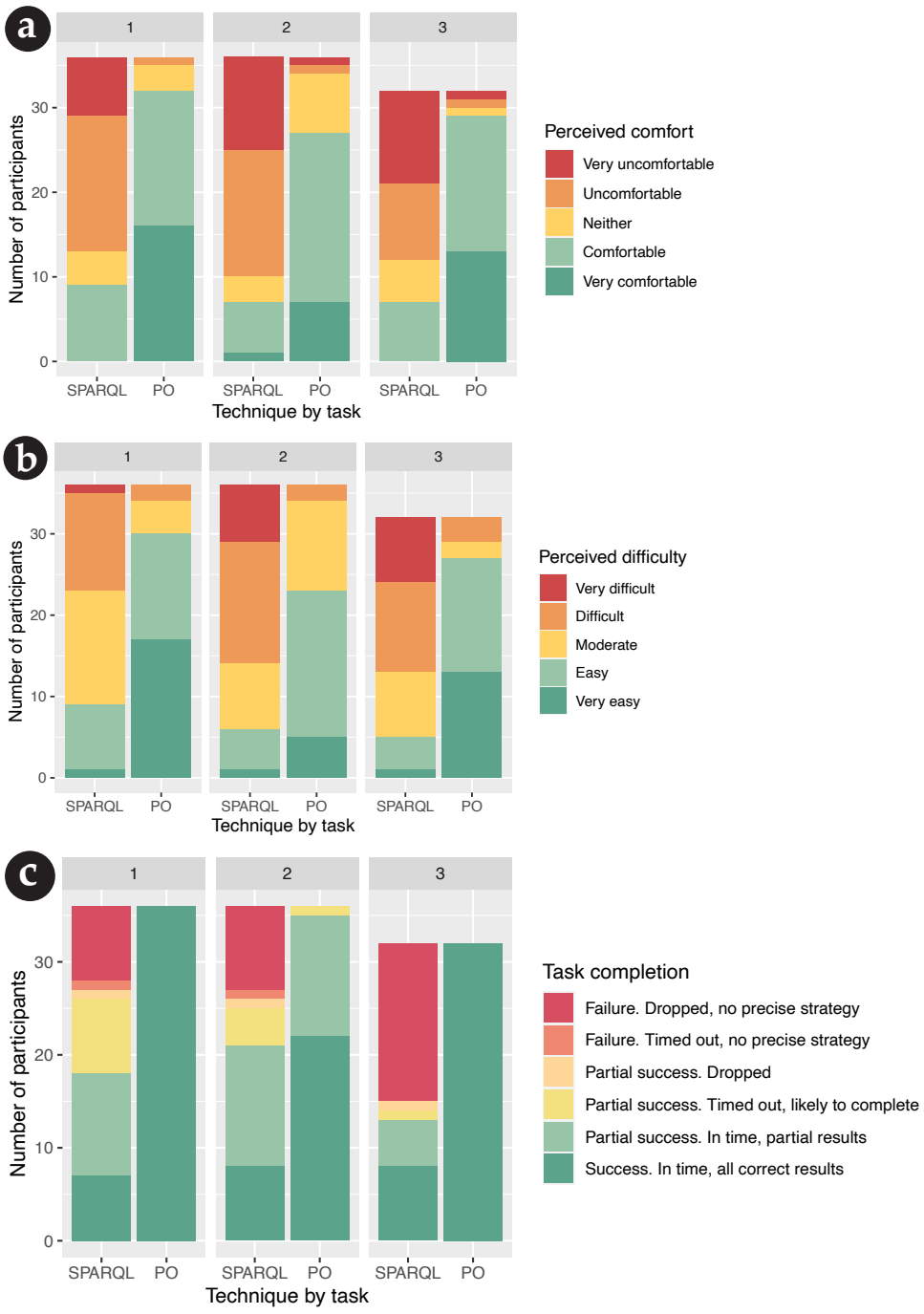


Figure 42: Comparison of *Path Outlines* (PO) and SPARQL-V (SPARQL) on 3 tasks. a) and b) are on a Likert-Scale. a) Participants find *Path Outlines* more comfortable, b) they perceive similar tasks as easier when performed with it, c) they are abler to complete the tasks successfully with it.

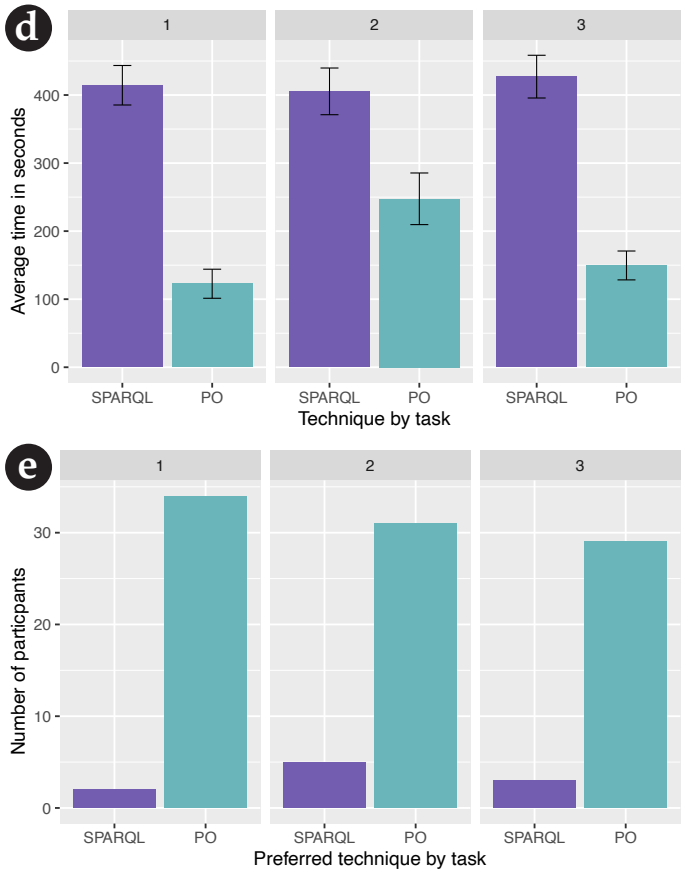


Figure 43: Comparison of *Path Outlines* (PO) and SPARQL-V (SPARQL) on 3 tasks. d) Participants are quicker with it and e) prefer it to SPARQL-V.

complicated. Another error which happened 5 times was that the query timed out, although it was correct. There are tricks and workarounds, but in most cases, the time needed to write the query and realise it would time out was already too long to start figuring out a workaround. This is a common problem with federated queries on sets, also reported by Warren and Mulholland [131]. Overall, our results are in agreement with H3.

4.7.6.2 *Memorising the main features of a dataset*

At the end of the experiment, participants answered MCQ questions about the structure of both datasets. Answers were very sparse, most participants did not remember the information at all, and there was no significant difference between the techniques. We cannot make any conclusion from the data we collected. We think this is related to the fact that participants were entirely focused on finishing the tasks in time, and did not have time to look at contextual elements of the interface. Therefore, the results are not in agreement with H4.

4.7.6.3 *Preference*

Most participants preferred *Path Outlines* (34 on T1, 31 on T2 and 29 on T3) versus Virtuoso SPARQL query editor (2 on T1, 5 on T2 and 3 on T3), as shown in Fig. 43b. In the comments collected at the end of the experiment, several participants spontaneously expressed their preference again: “Would definitely prefer to use the tool to explore the model”, “So, I would definitely prefer to use the tool over SPARQL”.

4.7.6.4 *Other user comments*

Several participants expressed the need for such a tool as *Path Outlines* in their work and asked if they could try it on their own data. Most of them liked the tool and made positive comments. One participant wrote an email after the experiment to thank us for the work, saying that “such tools are needed due to the conceptual difficulties in understanding large complex datasets”. It is interesting to note that the participant happened to be one of the two participants who had difficulties to relate to the tasks during the interview.

Our tasks are challenging to perform with SPARQL because they need to be decomposed in many steps, combining several types of difficulties, and they require to think in two dimensions: broad to consider sets of entities and objects, and deep to traverse the graph. This is not intuitive, and the cognitive

load to remember the sequences of a path is heavy. Our tool only required to browse and select, as it used the granularity required by the task.

4.8 DISCUSSION AND CONCLUSION

RDF data producers face a challenge: the particular structure of their data questions the efficiency of traditional summarisation and visualisation techniques. To address this issue, we presented the concept of *path outlines*, to produce path-based summaries of RDF data, with an API to analyse them. We interviewed 11 data producers and confirmed their interest. We designed and implemented *Path Outlines*, a tool to support data producers in browsing path-based summaries of their datasets. We compared *Path Outlines* with SPARQL-V. *Path Outlines* was rated as more comfortable and easier. It performed three times faster and lowered the number of dropouts, despite the fact that participants had, on average, 5 years of experience with SPARQL versus 5 minutes with our tool.

We used coordinated views combining new visualisations with filter and detail panels to support the representation and manipulation of those complex objects. A limitation of our combination is that it relies on splitting the paths by depth. While this enabled us to display very high numbers of paths, there are cases where users would prefer to see several depths at the same time, as for Task 2. With the current interface, this means repeating the same task with different depths. In future work, we would like to investigate solutions to go from one depth to another more easily, and/or to inspect several depths at the same time.

The concept of *path outlines* can be developed to support a wider range of metrics, such as the detection of outliers suggested by a participant in the first study. To go further in this direction, integrating statistics with the content [90] could make the path overview the entry point for an iterative analysis of the content, as advanced profiling tools in other databases communities start to do [62]. This would allow to profile subsets of entities and address more elaborate tasks, such as finding the reasons for a problem pointed out by the summary. Supported by other visualisations, to be designed and implemented, the concept can have many applications. For instance, it can support ontologists in bettering the quality of RDF data models, showing how a modification of a property in the model would impact the potential paths traversing it, addressing their needs to “make changes to the inferred hierarchy explicit” [127].

We believe that the development of Knowledge Graphs will benefit from path-based summaries and tools such as *Path Outlines*, presenting information to users at a granularity and form matching their need to make sense of information. We think that such tools will help overcome some of the complexity due to atomising data as RDF triples, and leverage high-quality Knowledge Graphs.

5

THE MISSING PATH

This chapter is a revised version of two papers co-authored with Jean-Daniel Fekete: a full paper accepted for publication in Sage Information Visualization journal [26], and a workshop paper [27]. My contribution included the initial idea, the design of the system, its implementation, its evaluation, and writing most of the paper. Jean-Daniel Fekete contributed to shaping the idea and supervised the design, the evaluation, the implementation and the writing of the paper.

The Missing Path is a tool to identify and analyse incompleteness related to groups of entities in Knowledge Graphs. It relies on a map of entities based on their completeness, combined with detailed statistical summaries of their *path outlines*. The tool is available as open source at gitlab.inria.fr/mdestand/the-missing-path and can be run online at missingpath.lri.fr.

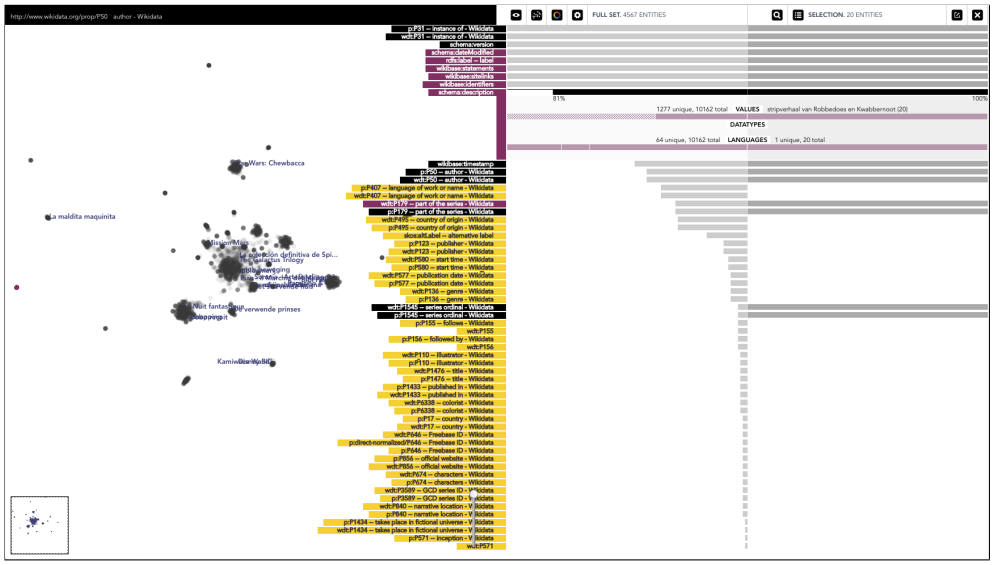


Figure 44: The map on the left shows the 4567 entities of type `wdt:Q1004 Comics` in Wikidata. The clusters appearing represent groups of entities that share the same missing paths. The user has selected a small cluster of 20 entities on the left of the map; it is coloured in dark pink. On the left column, the histogram of paths completeness for the full collection can be compared with the histogram for the selected subset on the right. Each row represents a path as a grey bar; its length is mapped to its percentage of completeness. The left part of a row is coloured in yellow if the path is missing in the selected subset and in dark pink if there is a significant difference between the full collection and the subset summaries.

5.1 INTRODUCTION

Knowledge Graphs (KG) allow to merge and connect heterogeneous data despite their differences, and this flexibility is key to their success. As they are incomplete by design, the drawback is that entities in a collection can have heterogeneous descriptions, potentially producing unreliable query results. Data producers, people, and organization producing KG data, still need to ensure, as far as possible, the best level of completeness. Completeness is regarded as an essential criterion in most quality methodologies for RDF data, the most used framework to describe KGs [46, 75].

The difficulty is that they have no means to distinguish cases where incomplete entities can and should be fixed. Let us consider the example of a publishing company building a KG with the books they publish and related books, such as those which inspired their books or are quoted in them. The graph merges data coming from several databases in the company, regularly enriched with information gathered from external sources such as Wikidata [78] or Geonames [42]. Connecting their data in a KG enables them to power recommendation algorithms, to running analysis of the sales, and so on. Sharing this KG also allows researchers in humanities to analyse the books, and libraries and resellers to reuse the metadata. However, if the data are incomplete, such applications may lead to erroneous results, such as wrong decisions based on the incomplete analysis. While allowing incomplete information makes it possible to merge the various sources, some of the missing data might be fixed, but the various strata of data that were added and modified at different points in time make it difficult to identify them.

Available tools and methods assess a completeness rate to each property in a collection and produce flat lists of all entities missing each property. Once assessed that, for instance, the publication date is missing for 11% of the books (1346 books), the data manager has to inspect one by one a list of 1346 entities. After uncountable hours and thousands of clicks, she will maybe find out that certain issues were recurrent, and that she could have fixed them in bulk. She might realise that more than a hundred books came from the same original database describing the related books and that the date was actually not missing in the original data, but happened to be an uncertain date, expressed by a year, followed by a question mark (e.g. ‘1943?’). She might also notice that several of them had been published during war periods, while another significant part of them had been published clandestinely. Finding what those subsets had in common at the start would have given her a very useful hint: it was very unlikely that she would find more precise information by

looking for the date in external data sources; she could have spared hours of unsuccessful research. Other subsets of interest could include books planned for publication, which can only be fixed later, when the date is known; or all books from a specific source, pointing to a bug in the transformation process, in which case she would rather fix the bug and run the transform script again rather than fixing entities one after another; and so on. She might also never notice those facts, as it is very difficult to find the coherence of scattered items when inspecting them in random order, especially if there are many meaningful subsets.

Our tool, *The Missing Path*, aims at addressing this issue. The map, grouping entities according to their incomplete profile, lets users identify consistent subsets. Comparing a specific subset with the full collection reveals its distinctive features, giving useful hints to understand the cause of incompleteness, and fix entities in bulk, saving significant time. *The Missing Path* considers the completeness not only of direct properties (e.g. the publisher of a book) but also of indirect properties (e.g. the location of the publisher of a book), also called *paths* of properties. The novelty of our approach is 1) to use a map to identify structural similarity of entities in a KG, and 2) to support comparative analysis of the distributions of values at the end of paths of properties in a KG.

We first introduce the basics of RDF and discuss related work regarding the evaluation and the visualisation of completeness. Then, we present the tool; we describe how path-based summaries are extracted and computed, we explain the design rationale and the main parts of the interface, and we illustrate it with a use case featuring a fictional Wikidata contributor. Eventually, we relate the iterative design process we used to improve and validate our approach while working with nine Wikidata contributors, following a methodology inspired by the “Multi-dimensional In-Depth Long-term Case Studies” (MILCS) of Shneiderman & Plaisant [115].

5.2 BACKGROUND AND RELATED WORK

We introduce RDF and we discuss related work regarding the assessment and visualisation of their completeness.

5.2.1 *Introduction to RDF data*

RDF data are graph data; their power relies on their structure: they are made of low-level statements, named triples, that can be chained to answer complex queries, possibly over several data sources. `example:AuthorA schema:author`

`example:BookB` is a triple, stating that Author A is the author of Book B. Triples are composed of a *subject*, a *predicate* and an *object*. *Subjects* are always *entities*, represented by URIs. For readability, URIs can be prefixed: `example:AuthorA` stands for `<http://www.example/AuthorA>`. *Predicates* — also named *properties*—also URIs; they follow rules defined in domain-specific models named *ontologies*. `Schema.org` is an ontology specialised in the description of web pages, and `Schema:author` is one of the properties defined in it. *Objects* can be *entities* or *literals*. When an object is an entity, it is possible to chain statements, for instance: Author A is the author of Book B, Book B's publisher is Editor C, Editor C's location is City D, City D's name is 'Paris'. The chaining stops when the object is a literal, like `'Paris'`, since a literal cannot be the subject of another triple. A chain of predicates is named a *path* in the graph.

The RDF framework is very flexible and allows each entity to be described with different properties. However, to make their data meaningful and usable, data producers need to ensure a minimum of homogeneity.

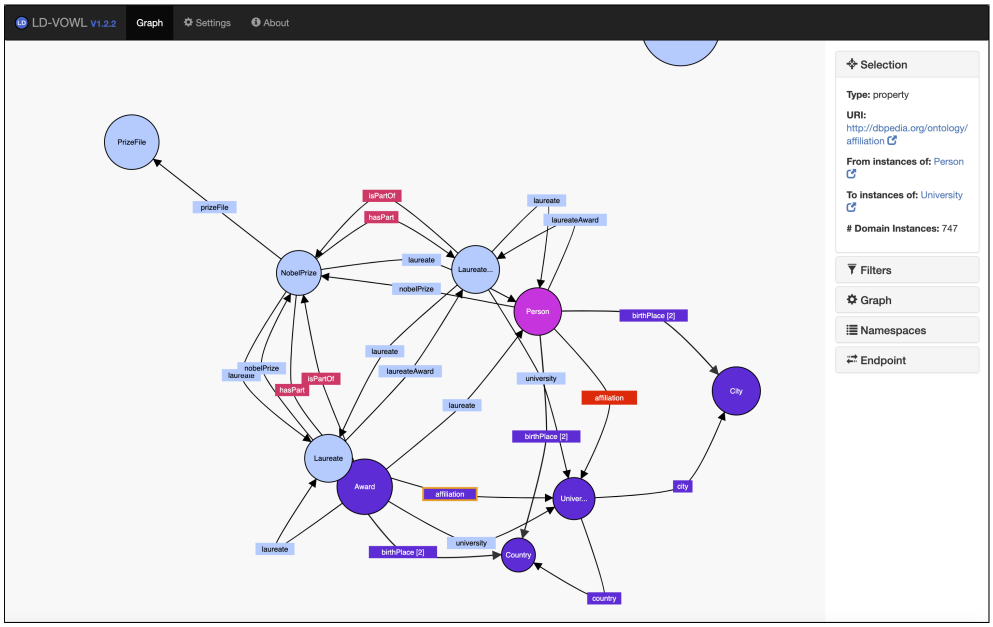


Figure 45: Screenshot of LD-VOWL, taken on 2020-12-12 at vowl.visualdataweb.org/ldvowl. The user has selected the property 'affiliation' (in red) and can see in the top right panel that it is used 747 times. To know the rate of completeness of this property relative to the class Person, she needs to select the node Person, read in the panel that there are 910 instances, and compute that 747/910*100 = 82% of the persons have an affiliation.

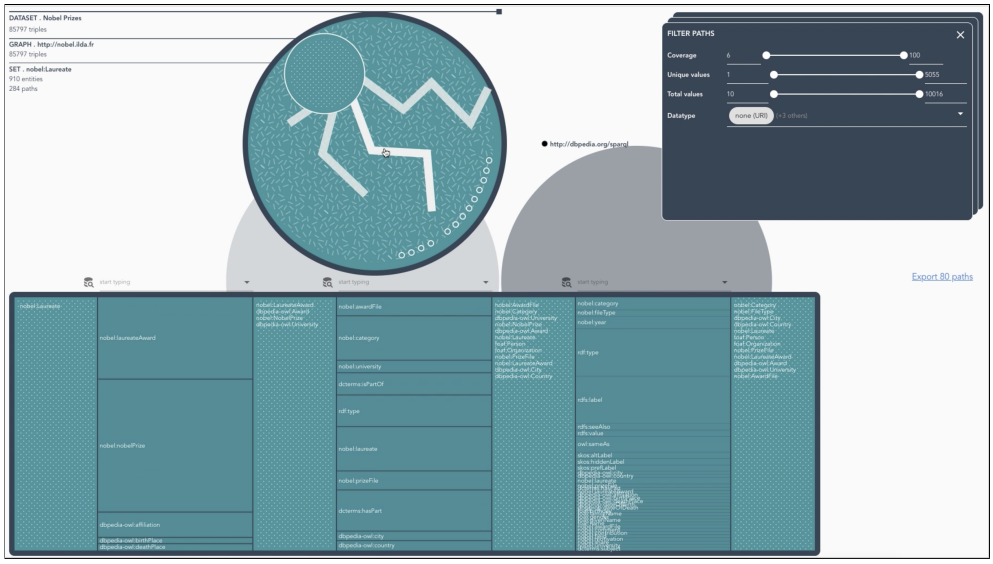


Figure 46: Screenshot of Path Outlines, taken on 2020-12-05 at spf.lri.fr. The user can browse the paths for a collection, filtering them on their completeness rate (among other metrics), and inspect the completeness rate of each path.

5.2.2 *Completeness in RDF*

Though the definition of quality in RDF can have many acceptations, most work on the topic mention completeness as important criteria [14, 17, 75, 100, 138]. The rate of completeness of a property is the percentage of entities in a given set described by this property. The set of entities can be the dataset or a subset. Technically speaking, approaches considering the dataset [11] give the most accurate overview. However, from an editorial point of view, and except for some very generic properties, like `rdfs:label`, that might apply to any entity in a dataset; it is more reasonable to expect homogeneous descriptions for groups of entities that are similar, also named collections of entities. Issa et al. [61] use the class of resources as similarity criteria, for instance `schema:Person`, `schema:Organization`, or `schema:Place`, and display the result as a UML class diagram. They do not support the evaluation of the completeness of paths of properties. A typical use case would be to evaluate the percentage of authors whose place of birth has geocoordinates, to know if plotting a map would give a representative overview of authors. Using a node-link diagram to lay out a summary graph of the dataset allows to read paths of properties [120, 132], as displayed in Fig. 45, with the limitation that the counts are given as absolute counts for each selected element. The user has to compute the rate himself for single properties, and cannot access it for paths of properties. To address this limitation, Path Outlines lets users browse

paths following their completeness rate and other metrics [28], as displayed in Fig. 46.

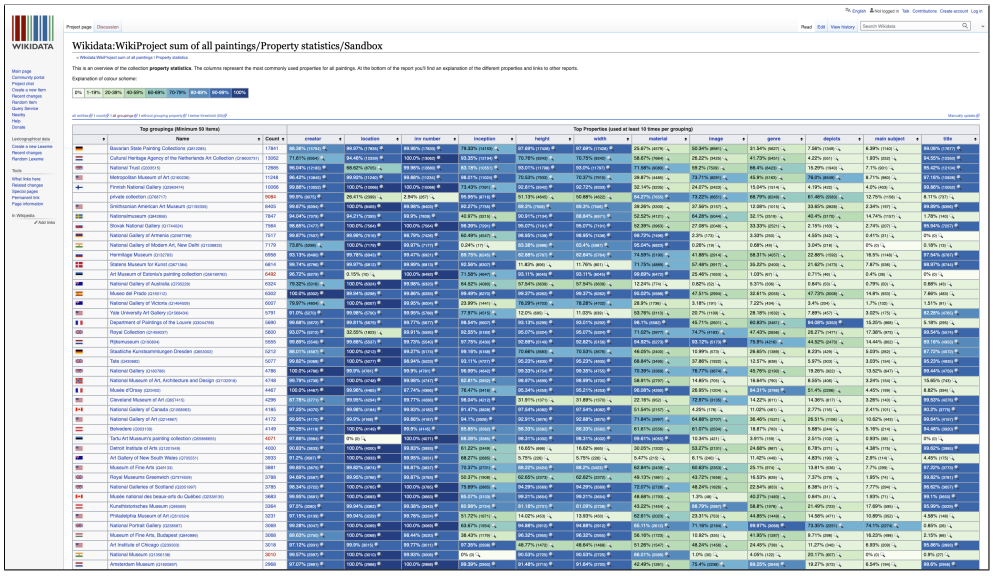


Figure 47: Screenshot of Integraality for Wikidata, taken on 2020-12-12 at wikidata.org/wiki/Wikidata:WikiProject_sum_of_all_paintings/Property_statistics/Sandbox. The color scale helps users compare the completeness rate in the different groups. However, as the table scrolls over more than 5 screen heights, it is actually difficult to read and use.

However, considering the rate of completeness of a property or a path of properties relative to the full collection might not always be enough to help data producers fix their datasets. In RDF, meaningful aggregation can also be achieved through the values of a property. For instance, entities in the collection `schema:Person` could be considered regarding their profession, encoded in the value of `schema:hasOccupation`. This allows identifying smaller subsets with similar profiles and needs. Integraality [118] lets users select a property to define subsets in the collection, and then evaluate the completeness of other properties relative to those subsets, as displayed in Fig. 47. The limit is that the table can be huge and thus difficult to read and use and that one single property might not produce useful groups to analyse the completeness of all properties. PRO-WD [134] supports crossing several properties, but produces a grid of charts that is very difficult to interpret.

Our approach, instead of starting from the values at the end of properties to define consistent groups, identifies clusters of entities with a similar structure, in order to automatically reveal meaningful contexts concerning the properties over which completeness is evaluated.

5.3 DATA REPRESENTATION AND PROCESSING

The originality of our approach relies on the data representation. We build on the concept of *semantic paths* to summarise the description of a collection, and we use the semantic paths as indexes for vector embeddings to compute a map of completeness as well as detailed summaries.

5.3.1 *Paths summaries*

We build on the concept of *semantic paths* to describe a given collection; they encode aggregate information relative to chains of triples. In the article introducing them [29], their description is limited to the counts of unique and total values at the end of the chain. We use vector embeddings to extend them and offer a detailed summary of their distribution. Our API takes as parameters *the URI of a SPARQL endpoint*, *a similarity criterion for the collection*, *a maximum depth for the chains of properties* to analyse. We extract RDF data to process them into a matrix. We first retrieve all the path patterns—the combinations of chains of properties that will be analysed—up to the max depth, and their completeness rate, as described in [28]. The list is ordered by completeness, starting with the most complete path, and stored in a file. We assign an auto-incremented index as an identifier to each path.

5.3.2 *Retrieval of the entities*

Then we fetch the URIs of all entities in the collection. A specific issue with SPARQL endpoints is that an endpoint cannot return more than a given number of lines as a result. This `quota` is usually set to 10,000 by default. Unlike SQL databases, there is no guarantee to retrieve all the results repeating the same query using the LIMIT, START, and ORDER BY commands. We use the *semantic paths* to find a path to formulate several queries so that each query will retrieve less than `quota` entities. We initially set a `maxUniqueValues` variable to 30 in order to keep the number of queries reasonable. We start by checking the best-represented path with less than `maxUniqueValues` values at the end, and we retrieve the unique values at the end of this path, and the count of entities associated with each. If the highest count is lower than `quota`, and the number of entities not represented by this path is also lower than `quota`, we use this path to retrieve the entities: for each value, a query fetches all the entities having this value at the end of the path; then the last query fetches all entities not described with this path. We merge and deduplicate all the entities retrieved. We assign an auto-incremented index as an

identifier to each entity. The list is stored in a file. Otherwise, if the path does not match the requirement, we consider the next path. If none of the paths meets the requirements, we increase `maxUniqueValues` and check the list of paths again.

5.3.3 *Values as vector embeddings*

Each entity is described as a vector, where each column is a path. The value is either 'null' or a list of descriptors, structured as follows: `[values, datatypes, languages]`. Each element is itself a list, to account for multiple values, since cardinality is not constrained in RDF. For instance, a cell describing the label of an entity with 2 labels could contain: `[['À la recherche du temps perdu', 'In Search of Lost Time'], null, ['fr', 'en']]`. The datatype descriptors are filled only if they are expressed in the data. A cell describing the publication date of an entity could contain: `[[1998], ['xsd:dateTime'], null]`. The vector is stored in a dictionary, associated with the URI of the entity.

5.3.4 *The completeness matrix*

The matrix of completeness is created from those vectors, each row is an entity. The values are transformed as follows: 'null' becomes 1, meaning that a path is missing, and a list becomes 0. Then, we project the vectors in 2 dimensions, to be later used as coordinates on a map.

Dimensional reduction techniques [82] allow computing clusters and lay them out on a map. They usually group entities according to the values of their core attributes (for instance, the topics of a set of books and their publication date), to have items with similar descriptions grouped together [4, 139]. We fill the vector with the structure of the description; we consider entities as similar if they are described by the same paths of properties, even if the values at their ends are not the same, to identify groups of entities missing such information. Among the large number of dimensionality reduction techniques available [82], we opted for UMAP [74]; this flexible method accepts both simple or sparse vectors—as we knew that the number of paths to consider, that is, the number of dimensions in a vector, could vary significantly across datasets—, and is fast and efficient for clustering. We use UMAP with the dissimilarity function *Russel-Rao* from the Scipy library [110]. This function computes a dissimilarity that takes into account the indices of the Boolean values in the vector—as opposed to a Jaccard function, for instance. As a result, items that form clusters on the map are those missing the exact same

set of paths—while Jaccard would have grouped entities missing the same number of paths. To our knowledge, using maps to identify similarities in KGs is a novel approach.

5.3.5 *Advanced summaries*

To produce the summaries, we construct a matrix with all the vectors, and we transform it into a table (a Pandas DataFrame) to compute the summaries with Python.

The summaries are based on unique values. All values with a number of occurrences lower than 5% of the total number of values are merged in an 'other' bucket to keep the overview readable. The graphical elements can be used to select entities by clicking on them, as displayed in Fig. 52. The 'other' bucket can also be used as a selector, and its values will be detailed as the selection narrows down.

To detect statistically significant differences, the system uses the distributions of the values at the end of a path for the subset, and for the full set, as displayed in the summaries, including the 'other' aggregate. It normalises them and compares them against each other, performing a Kolmogorov-Smirnov test, using the `scipy.stats.ks\_2samp` Scipy function. It then repeats this operation with the summaries of the datatypes and languages. If there appears to be a significant difference (p-value < 0.1) in either values, datatypes, or languages, the path is colored in pink.

5.4 USER INTERFACE

This new data representation allows us to design an interface to analyse the incompleteness of subsets in RDF data (Fig. 44). We will present the design rationale and detail the main parts of the interface: the map, the histograms with embedded stacked charts, and the selection bar.

5.4.1 *Design rationale*

To support the identification and analysis of subsets of entities relative to the completeness of their paths, The Missing Path coordinates an entity-centric visualisation, the map, with a path-centric visualisation, the histogram. The map represents all the entities in the collection and allows to situate a selected subset through explicit color encoding (in pink) of selected items. The path summaries describe the full collection and the selection and are laid out in mirror. The combination of *superposition* (on the map) and *juxtaposition* (in mirror) allows the effective support of comparison [44]. The tight integration of

statistics and visualisation is known to support explorative data analysis [90], helping users to make sense of the data.

There are two ways to select a subset of items sharing a similar structure: selecting a cluster on the map or using a combination of graphical elements in the summarised distributions to express logical constraints, e.g. *all items missing pathA and pathB, but not missing pathC*. The map is intended to guide users in their discovery, while the summarised distributions help them to refine a selection, or to fully express their own constraints to pursue their ideas when new ideas come to them [91].

5.4.2 *2D map of entities*

On the left part of the screen, the map (Fig. 44) displays clusters of items with similar incomplete profiles, offering an overview of the entities in the collection and allowing to select the clusters. It supports the following tasks:

- see the homogeneity of the collection, regarding the completeness of the paths selected to compute the projection;
- select subsets, through precomputed groups or using the interactive lasso; and
- identify entities that are selected.

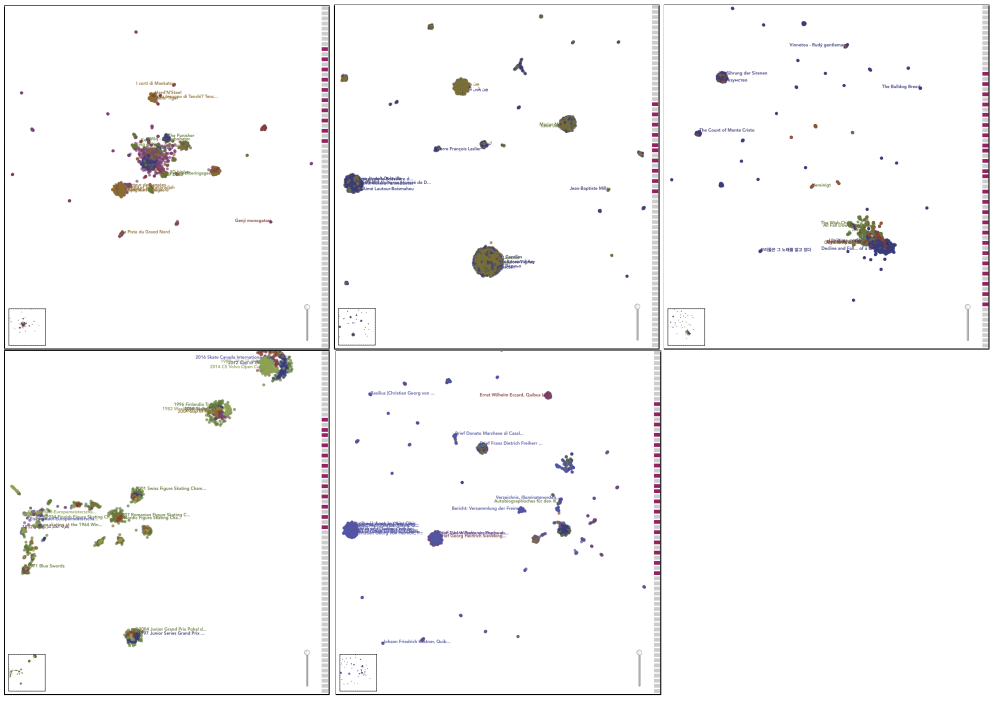


Figure 48: Collections C1, C2, C3, C4 and C6 (see Table 4). The number of clusters, their size and distribution provide a visual footprint of the shape of a collection, relative to the set of paths selected to produce the map (highlighted in pink on the right side of each thumbnail).

5.4.2.1 *Overview of the completeness*

Fig. 48 shows that different collections have different footprints. If a collection were 100% complete, there would be only one large cluster. The number of clusters, their size, and distribution, form a visual footprint giving the shape of a collection relative to the set of properties selected to produce the map. Users can modify the list of paths taken into account to build the vector with the projection button  and recompute the map. Our Python API, based on UMAP-learn [123], takes a few to 30 seconds to recompute the map for the collections in Table 4.

For instance, selecting only 2 properties, P1 and P2, to compute a map, could result in 4 clusters: entities missing both P1 and P2 properties, entities missing none of them, entities missing only P1 property, and entities missing only P2 property. The interest does not lie in the systematic enumeration of all combinations (in which case a table would be as efficient as a map). In reality, when more properties are taken into account, not all combinations happen, some are very frequent, and other concern only a few entities, and the map reveals unexpected clusters serving as entry points to explore a collection. Inspecting the profile of a cluster often reveals other similarities, that may relate to the provenance, the history, or the contributor.

5.4.2.2 *Colors*

While the position of the entities is based on missing information, their color is linked to the content of present information. Paths for which the summary of values has more than one value are candidates for color-coding. By default, the most covered candidate path is used. For instance, the default for collection C1 is `wdt:P31 instance of`, its summary is composed of two values: `wd:Q1004 Comics` and the aggregate `Other`. Entities are colored in blue for the former, in green for the latter, or with a gradient if they hold several values. Users can select another path to color the entities with the color button  in the top bar. When a subset of entities is selected, they are colored in pink and others in black.

5.4.3 *Paths histograms*

Next to the map giving a visual overview of the entities, the histograms (Fig. 44, right) offer an overview of the aggregated completeness of each path, for the full set and the selected subset. Stacked charts embedded in the histograms give access to the distribution of each path. They are laid out in mirror to let

users compare the profiles of the subset and the full set, in terms of completeness and distribution. They support the following tasks:

- see and compare the homogeneity of the full set with the selected subset, regarding all properties
- see and compare the completeness and distributions of the full set with the selected subset
- select entities based on the presence or absence of a property
- select entities based on summarised distributions of the values, languages, and datatypes at the end of the paths.

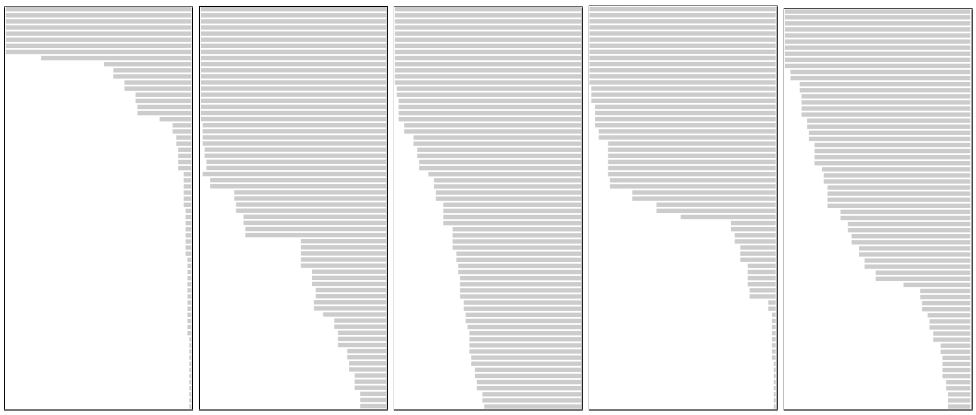


Figure 49: Collections C1, C2, C3, C4 and C6 (see Table 4). Histogram on the frontpage: the steepness of the curve gives a visual footprint of the completeness of the most complete paths in the collection. Scrolling down allows to see all paths. C1 is our demo collection, it was not curated as a wikiproject, so very few paths are fully complete, and there is a sharp decrease with a long tail of paths with a low rate of completeness. C2 is maintained by an active team of 10 contributors, a large number of paths is complete. C3 is more balanced, it is a catalog of films curated before it was imported. C4 has been created and curated over a short time mostly by one contributor. C6 is a starting project mixing sets of data which were curated separately.

5.4.3.1 *Overview of the completeness*

The grey bars represent all paths describing the collection, ordered by completeness, to give another visual signature of the completeness, showing at first glimpse the number of paths fully complete. Fig. 49 shows paths summaries for the collections displayed in Fig. 48. The map and the histogram are linked and coordinated.

Each row represents a path; the length of the grey bar is mapped to its percentage of completeness. Clicking on a path opens it, showing a summary as detailed in the next paragraph. Paths labels are displayed on the left of each row. By default, they appear when users hover a path, when they hover

a predefined zone on the map, as in Fig. 51, or when a path is open. Users can toggle them on permanently as in Fig. 44with the labels button .

5.4.3.2 *Summarised distributions characterizing a path*

When open, a path displays a summary of the distribution of the values at its end, as well as of their datatypes and languages. This work builds on the concept of *semantic paths*. Originally, their description was limited to the counts of unique and total values at the end of the chain [29]. We extend them by adding a more detailed summary of their distribution. Our API takes as parameters *the URI of a SPARQL endpoint*—a service accepting SPARQL query over an RDF dataset, *a similarity criteria for the collection*, *a maximum depth for the chains of properties* to analyse. We first retrieve all the path patterns—the combinations of chains of properties that will be analysed—up to the max depth, and their completeness rate. Then, to be able to compute summaries on any subset in a time that is acceptable for interaction, we retrieve the values at the end for all entities and store them in a matrix that can be processed rapidly with Python. We precompute a summary of the collection. The summaries of subsets will be computed on-demand.

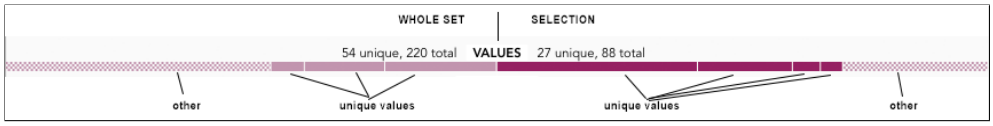


Figure 50: Summary of values for a path: the whole collection is presented on the left, in comparison to the selection on the right. The summary details values representing more than 5% of the total, and aggregates others: for the whole collection, only 3 of the 54 unique values are well represented enough to be detailed; the 51 that remain are merged in the 'other' rectangle, represented with a dotted texture. Hovering a rectangle displays the label and count of the value it represents. Each value, including the aggregate, can be clicked to be added as a condition for a selection.

5.4.3.3 *Comparison of the full set with the selected subset*

To make sense of a *subset* of entities, users need to identify its distinctive features, what defines it in comparison to the whole collection. The histogram is laid out as a mirror of the histogram for the full collection, to facilitate this comparison. For instance in Fig. 44, comparing the two histograms shows that the subset is very homogeneous; although it misses important information (no grey bar in the right column), the 16 paths that are described are complete (full grey bar in the right column), while only 8 of them are fully complete for the full set. Paths missing in the subset are highlighted in yellow, to help users focus on the problem they are trying to solve.

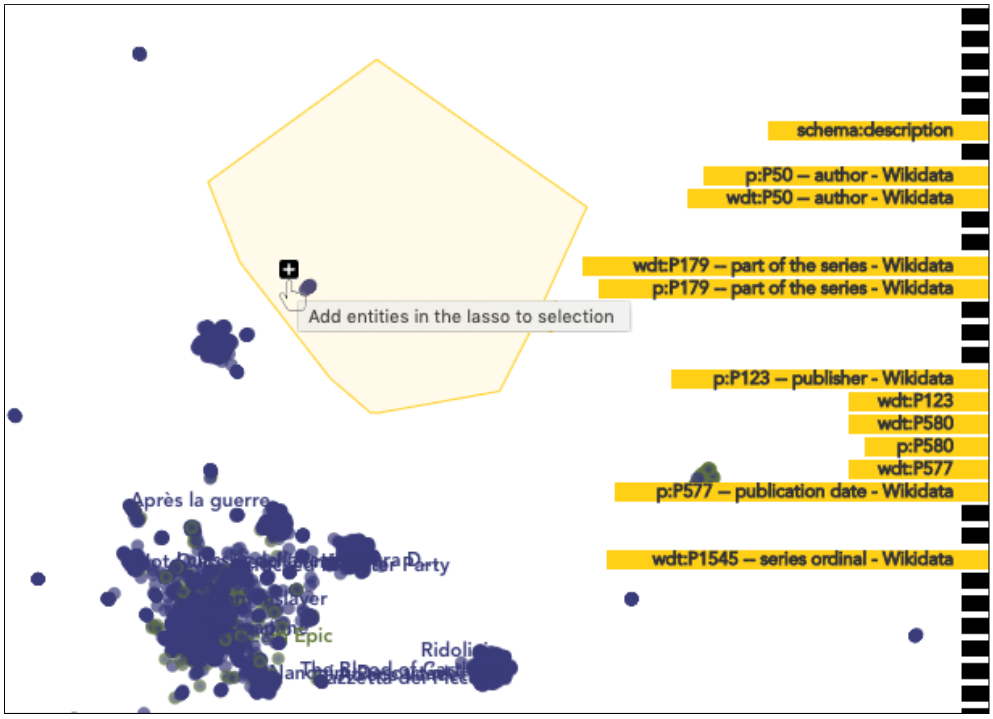


Figure 51: Hovering a predefined zone on the map highlights it in yellow, and gives access to the + button, to use it as a condition for a selection. It also displays and highlights in yellow the names of the paths missing for the entities in this zone.

Figure 52: The user can click on an element of the summary to add it to the selection (top). Once added, it becomes dark pink, and clicking again will remove it (bottom).

Figure 53: The selection bar contains controls to inspect and refine the conditions for a selection and its result. The number of checkboxes in (a) shows how many conditions are pending (here, there is one). Clicking on (a) displays the query in pseudo code (see Fig. 54). Clicking on (b) retrieves the list of entities matching the conditions and their summary. When a selection has been retrieved, (c) indicates the number of the list of entities in the selection, clicking on it displays the list in Fig. 55. (d) enables to export the selection, and (e) to clear it.

Figure 54: Conditions for a selection are expressed in pseudo code, to let users understand how the tool retrieves entities. They can refine them by toggling the elements that are underlined : 'having' can be switched to 'not having', resulting in the inverse condition, and 'the whole collection" to 'the current selection'.

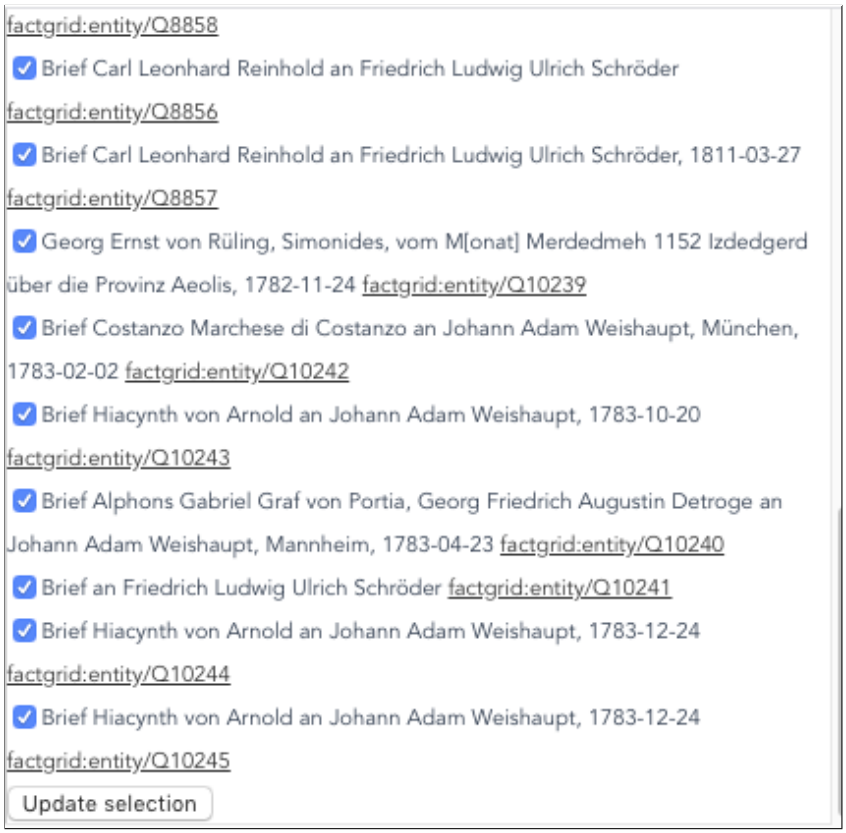


Figure 55: List of entities in the current selection. The label is in the preferred language when available. Clicking on the URI opens it in a new window.

To support users in the comparison task, the tool also draws their attention to which paths to inspect in order to understand the specificity of a selection (how it differs from the full set); it colors them in pink.

The yellow color indicates paths that are missing in the subset. It stands out, more intense and luminous than the other colors in the interface, to draw the attention of users to what is not there, and help them make sense of the absence.

5.4.4 *Selection bar*

The selection bar supports users in inspecting and refining the conditions for a query. *Conditions* are selection criteria in the database sense, combined by a conjunction (an “and” operator). Hovering over the map highlights predefined zones (Fig. 51). The + button in the centre of the zone allows adding the zone as a condition. Clicking on the map switches from region to lasso mode, to let users select zones that are not predefined. Graphical elements in the histograms and the summaries can be added to and removed from the selection. The selection control bar in Fig. 53 supports users in understanding what happens when they add a condition, validating the selection, seeing the list of entities selected, and clearing the selection.

a) *Toggle list of conditions*. Each condition is represented by a checked box. When at least one condition has been added, (a) and (b) become pink, to indicate that the selection can be queried. Clicking (a) toggles the list of conditions, as shown in Fig. 54. The query is written in pseudo-code; users can remove conditions from the list, toggle them to their inverse condition, or toggle the scope of the query from ‘whole collection’ to ‘current subset’.

b) *Inspect selection*. The combination of conditions defines the selection. When users clicked the inspect button, the query is sent to our Python API. The new list of entities in the selection is retrieved, and Fig. 53-c is updated first. Then the summary for the entities is computed and displayed under the selection control bar Fig. 53-f.

c) *Toggle list of selected entities*. Clicking this button toggles the list in Fig. 55. Users can remove entities from the list. Clicking the ‘Update selection’ button at the bottom updates the paths summary for the selection.

d) *Export selection*. This button triggers the download of 3 `csv` files that can be used to keep track of the query: `condition.csv` contains the list of conditions used to get the selection, `selection.csv` contains the list

of entities in the selection (URI + label), and `summary.csv` contains the summaries for the subset and full set.

e) *Clear selection.* Clears the current selection and its summary.

5.5 SCENARIO OF USE

We designed our tool to help users see what is missing in their dataset and make sense of it. Let us describe the interface from the point of view of a contributor who wants to curate the Wikidata collection `Q1004 Comics`, describing comic books. She opens the tool, sees the map of entities in Fig. 44. As she moves the mouse, yellow zones delimiting clusters of entities appear, and paths that are missing for the zone are highlighted in yellow. Her attention gets caught by a small cluster, which misses many pieces of information that are important to describe comics, such as `P407 language of work or name`, `P495 country of origin`, `P123 publisher`, `P577 publication date` and `P136 genre`. She decides to inspect this group in more details: she adds this zone to the conditions for selection using the + symbol and validates the selection with the magnifier button. The selection bar announces a total of 20 entities, and the summary appears under it. Some of the paths are coloured in pink, indicating that their summary for the selection might be significantly different from the full set. The contributor hovers the paths highlighted in pink to see their labels and starts by opening `rdfs:label`. She notices that there are 20 distinct labels, all of them in French. Then, she inspects `schema:description`. Its summary reveals that a single value is repeated 20 times: “stripverhaal van Robbedoes en Kwabernoot” (“comic strip Spirou & Fantasio” in Dutch, a popular comic strip originally written in French). The 20 descriptions are in Dutch. She inspects `schema:dateModified` and sees that 20 entities were last modified on the same day. The `P179 part of the series` property indicates that 20 are part of the same series. She finds that those entities appear to have very similar needs. According to her quality standards, labels and descriptions should be available in similar languages (as opposed to labels being in French only and descriptions in Dutch only). From what she knows, Spirou and Fantasio comics are known enough that it should be easy to find the author, language, publisher, and publication date. The information can likely be found from the same sources for at least some of the albums. If She is lucky, one of the sources might even be the URI of the series that all entities belong to. It looks like she will be able to save time by fixing those entities at once. Now that she has identified that this cluster needs a certain type of action, she would like to make sure that she will check all the

entities belonging to the series, even if they miss slightly different information and are not in the initial cluster. To do so, she clicks on the value shared by 20 entities to add it to conditions for selection. She then opens the conditions and reads the query: “SELECT entities HAVING the value `wd:Q1130014` at the end of the path `wdt:P179` among the current selection”. She toggles the scope definition from “current selection” to “full set” and validates the selection with the magnifier button. The selection bar now announces a total of 35 entities, all part of the “Spirou and Fantasio” series. She clicks the export button and downloads the files describing this group for fixing it later.

She then hovers the next zone. The paths highlighted in yellow indicate that entities in this zone also miss similar important information, the main difference being that they have a `skos:altLabel`, but no attribute `wikibase:timeStamp`. Note that even if the properties discriminating two neighbour zones do not appear to be meaningful properties, this structural approach helps detect coherent subsets. In order to inspect the new cluster, she adds the zone to conditions for selection using the + symbol and validates the selection with the magnifier button. The new selection replaces the previous one. The selection bar announces 127 entities. 100% of them have a `P179 part of the series`, so she opens the summary for this path that is now coloured in pink, hoping that she can detect interesting groups. The summary announces 25 unique values, and 3 values stand out because they are well represented. Those values are URIs, and she hovers them to dereference them in the URI bar above the map; she sees the corresponding labels: “Sammy” (25), “Bobo” (21), and “Natacha” (14). The rest of the values are merged in an ‘other’ group (67). She clicks on the first value to add it to conditions for selection and validates the selection with the magnifier button. She exports this selection. She repeats the same actions with the two other subgroups. Now she can refer to the `csv` files she has exported to fix each of those 3 groups.

This exploratory approach enables her to quickly detect small groups that are coherent and thus easy to fix. Let’s now see how she can use the tool starting from the summary of paths. She clears the current selection and clicks on the eye pictogram to display all path labels. She figures out at first glance, from the length of the grey bars in the histogram, that less than half of the entities have an author. She decides to make this a priority to fix. She opens the author summary, which confirms a completeness rate of 42%, and she clicks on the bar to add it to conditions for selection. She opens conditions to read the query: “SELECT entities HAVING the path `wdt:P50` among the whole set”. She toggles the condition from ’HAVING’ to ’NOT HAVING’

and validates the magnifier button. The selection bar displays 1929 entities for the selection. The summaries for paths are mainly composed of 'other' values. Wondering how to deal with this huge list, she considers refining the selection by combining conditions. She sees the property `P3589 Grand Comics Database Series ID` in the list. She decides to inspect entities having no author but such an identifier, which might mean that the information about the author will be accessible. The subset counts 49 entities, which is indeed more manageable. She exports the selection; the workflow should be easy since the source is the same for all entities; it might even be automatable. There are still 1880 entities without authors. She tries another strategy, looking for entities which have a publisher but no author. The result counts 129 entities.

With The Missing Path, incompleteness can be explored starting from the map or from the summary and then switching between them to refine or expand the exploration.

5.6 USER STUDY: ITERATIVE DESIGN AND EVALUATION

Using a methodology inspired by MILCS [115] we worked with Wikidata contributors to validate our approach and iteratively improve the design of the tool. This methodology is optimised to evaluate creativity support tool, and analysing incompleteness is a task that demands creativity, with no established method or measure to assess its effectiveness. It relies on an acute knowledge of the data and the workflow underlying their creation and edition.

5.6.1 *Participants*

We recruited 9 Wikidata contributors (2 female, 7 male) via calls on Wikidata mailing lists and Twitter. 3 were based in France, 1 in Sweden, 1 in Germany, 1 in the Netherlands, 1 in Australia, and 1 in the USA. 4 of them used Wikidata in the context of their work, and 5 as volunteers. They were 30 to 59 years old (avg: 39.89 yo, median: 34 yo). Their experience contributing ranged from 6 months to 7 years (avg: 3.46 years, median: 4 years). They spent between 1 and 165 hours a month contributing (avg: 52.89 hours, median: 24 hours). All participation was voluntary and without compensation.

5.6.2 *Set-up*

The interviews were lead online through a videoconferencing system. We used an online survey form to guide participants through the first interview and to collect demographic information. Our tool was run on a web server hosted by the laboratory and logs were filed in a database on our server.

Figure 56: Participants to our evaluation, corresponding to expert persona: *data producers* and *data reusers*.

5.6.3 *Procedure*

5.6.3.1 *First interview*

After going through the informed consent form and collecting demographic information, the interview was guided by the following question: *1. Which Wikidata projects do you contribute to? 2. How do you decide which data you will update in priority? 3. Did it ever happen that you wanted to contribute and didn't know where to start? 4. Can you tell me about the last item you edited? 5. Do you propose items for others to update? How do you select them?* Then we gave a quick overview of the tool and asked participants if they would be interested in visualising a collection with it.

5.6.3.2 *Second interview*

We first shared our screen with participants to present the tool and its documentation. We demonstrated basic tasks on the Comics collection in a 5 minutes demo. Then participants took control, sharing their screen so that we were able to observe them. They registered their unique identifier in the

tool for logs and performed the same tasks on their own collections. We explained to them how to give feedback using Gitlab issues. These Issues can be of three types: feature, problem, and insight. We encouraged participants to use any other communication channel if they felt more comfortable with it, explaining that we would transform it into issues ourselves. At the end of the interview, we created issues to file the reactions we have observed during the interview.

5.6.3.3 *Follow-up*

We communicated with participants by email (and a mix of Twitter direct messages and email for one of them). We conducted an additional video interview with four of them, during which we assisted them with the use of the tool when needed.

We name our participants P1 to P9, according to their unique identifier. We logged a total of 298 actions attributed to our participants, distributed as follows: add a condition (46), remove from condition (20), retrieve subset (74), compute projection (21), clear selection (21), load collection (61), and selectColor (55). P1 had no logs at all — his web browser privacy settings interfered with our log collection mechanism, although he reported using the tool. Over 4 months we conducted a total of 22 interviews, with an average of 2.44 interviews per participant (median 3), and we received a total of 111 emails or Twitter direct messages, with an average of 12.33 messages per participant (median 11). We extracted a total of 78 issues. Only three were filed directly by a participant; we transcribed all others from the interviews (54) and emails (19). One participant dropped out after the first interview, and one after the second, without giving a reason.

We used a total of 12 collections during the study, as listed in Table 4. Comics was our demo collection. Each participant had an initial collection, and three asked for the analysis of an additional collection during the process. The one who dropped out after the first interview had no collection.

5.6.4 *Data collection and analysis*

We recorded the first interview. For the second and third interviews, we relied on our notes to transcribe issues right after the interview. We also transcribed issues from emails and messages we received. At the end of the study, we exported the answers to the form and the issues into `csv` files, and we tagged the type (one of *collection, feature, general comment, insight, problem*) and the status (one of *solved, not relevant, future work*) of issues.

ID	Description	number of entities	number of paths
C1	Comics	4567	401
C2	French deputies	14513	1350
C3	BFI movies	6666	985
C4	Ice Skating 1	2204	94
C5	Ice Skating 2	1377	70
C6	Illuminati*	7938	183
C7	Maps	142	109
C8	Monuments in France	48845	775
C9	Monuments in Brittany	4210	367
C10	Research institutes	235	353
C11	Swedish female sculptors	292	395
C12	Swedish photographers	760	739

Table 4: Data collections visualised in the tool for the evaluation, available in the demo instance. * The Illuminati collection comes from an instance of Wikibase, Factgrid

5.6.5 *Results*

We analyse the results with regards to usability issues and validation of the approach.

5.6.5.1 *Usability issues*

The iterative design process helped us solve usability issues. The most critical issue was the understanding of the map. In an earlier version of the tool, the interface emphasised information missing in a subset after its summary was retrieved. P3 stated that he found it difficult to understand which paths were missing. We decided to precompute default zones on the map and to display missing path names on hover to make the interface self-explanatory. We also added the yellow color to highlight what was missing. After this, users reacted much more positively to the map: “I understand now” (P2), “Now I understand it better” (P3).

A second issue was the difficulty to identify a distinctive feature in the selection. P7 suggested highlighting the paths for which the summary appears to be significantly different in the subset than in the full set. In an earlier version, inspecting a cluster to understand its specificity necessitated looking at each path one by one, which was long and uneasy. Participants did not know where to start, and it could happen that they repeatedly opened paths for which the summary consisted in ‘other’ aggregates, which was not much help to identify the specificity of a group. We added the automatic detection of significant differences, as described in section Comparison of the full set with the selected

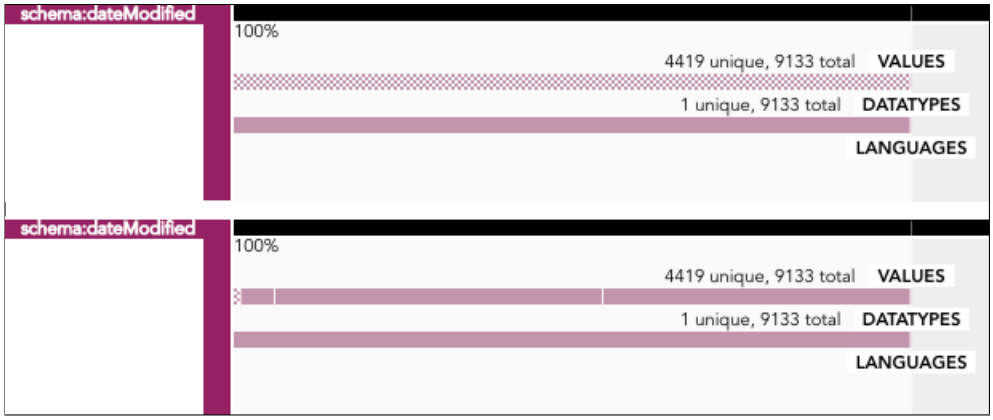


Figure 57: Evolution of the layout for dates summaries during the iterative process. This is the summary for the path `schema:dateModified` on the collection `C1 Comics`. In the first version (top) the dates were grouped by unique values, which very often resulted in an 'other' aggregate, laid out with a dotted texture. After participants' feedback we implemented binning for dates (bottom), which results in 4 groups, from right to left: "2018" (4150), "2019" (4423), "2020" (460) and 'other' (100) — hovering the rectangles reveal the value and counts. Each value can be used as a condition for selection.

subset, and highlighted them in pink. This feature saves substantial time and provides guidance.

The way we presented summaries also evolved during the process. We had first designed summaries for integers as boxplots, thinking it could be interesting for users to select only outliers or median. We realised that our users could not read boxplots and ignored those summaries, so we switched to a stacked chart of unique values, similar to the one used for text values. On the other hand, dates and times were initially designed as a stacked chart, and most of the time resulted in a single 'Other' aggregate. P1 asked if we could group dates, so we implemented binning into hours, days, months, or years. This feature improved the usability of some path summaries like, for instance, for instance, the modification date, as we can see in Fig. 57. When he first used the tool, P1 also tried to select the 'other' aggregate as a condition for selection, which was at the time not possible. We also added this feature.

All in all, participants suggested 32 new features and reported 15 problems. We implemented 20 of the new features, marked 3 as irrelevant in the context of our work, and kept 9 for future work. We solved 13 problems, marked one as an exception, and one for future work.

5.6.5.2 *Validation of the approach*

We were particularly interested in knowing if users would rely on it to start the exploration of subsets. P1, P9, and P2 did. P1 explained: "I see it as a way

to start the exploration, see the outlines". He had already spent a lot of time curating this set of data and knew them well. However, there are more than 14,000 entities in the set, and he worked more specifically on those related to the French Fifth Republic, so the map was useful to spot problems he was not aware of. For instance, the first cluster he inspected during the second interview was a set of 47 deputies having no place of birth. He commented: "There should not be entities with no place of birth. This group can easily be fixed, the information is available through the Sycomore French deputies database, and they all have a Sycomore ID" (`wdt:P Sycomore ID`). During the third interview, another cluster showed entities (deputies) with no given name. He explained: "All deputies should have a given name. This can be fixed easily from the labels." He thought that even if the focus might switch from the map to the histogram as you get to know your data better and they become more homogeneous, there can always be new stages when you incorporate new sets of entities and want to bring them to the same level of quality as the rest of the data when the map could prove to be useful again.

P9 did also start from the map. He was planning to import and manage his own catalogue of movies in Wikidata. Since he was still at a planning step, we had selected the BFI movie database, which was about similar in size and type of information to what his own data would later be. He figured out there was a cluster of 16 entities without titles. He inspected the summary and found out those entities all had a label, which meant the titles would be very easy to fix. A double-check through the histogram showed that there were 125 entities with no title but a label. Another cluster had no directors. This leads him to use the histogram to look for all entities having no directors, which amounted to 1380 entities. Looking at the map, he could see they were spread into about 20 different clusters, depending on what else was missing. Hovering the clusters then gave him an overview of the possible combination of missing attributes. He inspected two of them in more detail. Trying to imagine how he could use the tool later with his own data, he said he would probably want to configure the projection with paths he wished to achieve full completeness for, and then work on the data until there's only one big cluster.

P2 needed to customize the map, using only the paths that were of prior importance for him to compute the projection. This reduced the map to a few clusters that he found meaningful. "Now I am satisfied. This is the image I wanted when all the irrelevant criteria that complexified the map have been removed." Then he started his exploration from the histogram. He used the combination of conditions to find the list of all monuments qualified as churches

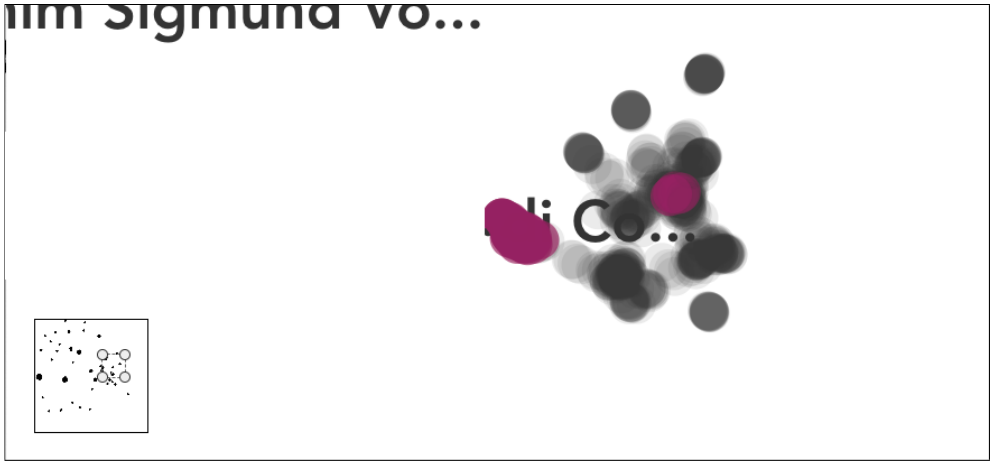


Figure 58: Entities highlighted on the map of the collection C6 when all entities having a `factgrid:prop/P17 Dataset complaint` are selected. The contributor who made those statements explained he worked on small groups of consistent entities, and we can see they appear as such on our map, although P17 is not used to compute the map. This shows that those consistent groups miss the same well represented attributes.

— having `wd:Q16970 church building` as a value for `wdt:P31 instance of` — but with no identifier `wdt:P3963 Clochers de France ID`, specific to churches. He expressed the wish to see the entities highlighted on the map, a feature described in section Colors, that we added following his demand. While explaining that `wdt:P18 images` was not a relevant path for the projection in his opinion, because it was normal that some entities had no images, he exclaimed "I know what I am going to do this afternoon!" He had figured out he could select all the entities having no `wdt:P18 image` but a `wdt:P373 Commons category`, because if they had a Commons identifier, then he knew he could find an image. He added "I could have done the same with SPARQL, but I would never have had the idea. The tool gave me the idea."

P5 preferred to start from the summaries and ignored the map. She suggested a feature to support better exploration from the summaries: the possibility to combine conditions to refine the selection. Interestingly, this made our tool much more flexible, able to support more diverse tasks.

In total, participants made 16 general comments on the approach and reported 12 insights on their data.

In summary, our study helped us make the tool more flexible and adapt it to different workflows. We had first thought the map would be the main entry point, and statistical summaries would help refine and analyse the clusters. We realised that looking at the histogram overview did also trigger ideas of specific completeness profiles (e.g. entities missing a specific path but not

missing another one, or entities with a specific feature and missing a path), which is another way to detect coherent clusters. The full list laid flat triggered associations that could be quickly verified.

5.7 CONCLUSION AND FUTURE WORK

We have presented *The Missing Path*, a visualisation tool to support data producers in analysing incompleteness in their data to identify subsets of items that can be fixed, based on two novel representations of RDF data. The map provides a structural snapshot of a collection, reflecting its history and allowing users to untangle its various strata. The histograms and stacked charts laid out in mirror allow comparing a subset with the full collection, revealing its distinctive features. The coordination of those new visualisations supports users in the interactive exploration and analysis of incomplete subsets. Our user study confirmed that Wikidata contributors could gain new insights and identify groups of entities that can be fixed. Participants guided us to make the tool more understandable and usable. Doing so, they also lead us to make it more flexible, supporting various workflows, and this pushed our tool in the direction of an exploratory analysis tool.

To our knowledge, there is no such tool for RDF data. In the future, we would like to investigate other analysis scenarios, besides incompleteness. We will also address the need to keep track of the various levels of understanding provided by the tool—not only by exporting the data, to let users monitor the evolution of their dataset.

Having heard of our tool, Wikidata product managers became intrigued, interested, and asked for a demonstration. As one of them told us when we demonstrated the tool, “One of the big problems our contributors face in keeping the data quality and completeness high is the fact that it is very hard to see the big picture due to Wikidata’s modelling being centred around individual entities. Your tool is addressing this issue”. We will continue to interact with the Wikidata community and other RDF data producers to improve our tool and support better quality Knowledge Graphs.

6

CONCLUSION AND DISCUSSION

6.1 SUMMARY

In this thesis, I investigated path-based methods to enable interactive exploration of Knowledge Graphs, from overview to detail.

In *Chapter 2* BACKGROUND AND RELATED WORK, I first introduced RDF data, explaining that their high granularity, while being very powerful and flexible, makes it difficult to reconstitute meaningful pieces of information. Then I presented 3 types of users who need to browse RDF data: *data producers* and *data reusers*, and *lay users*. I reviewed generic tools to browse and visualise RDF data, regarding their ability to browse from overview to detail.

In *Chapter 3* S-PATHS, I presented a semi-automatic exploration system aiming at providing meaningful overviews for any subset in a collection. Drawing on the fact that relevant information about an entity might be several triples away from it, I relied on a statistical analysis of the datatype and completeness of chains of properties of various lengths to describe a collection. A matching algorithm then compared those statistical objects with the requirements of a set of views to present the most readable overview to users. Users could select subsets to progress through the dataset or switch focus to other sets of entities. I reported a qualitative study showing that users can make sense of such overviews and remember the important dimensions of a dataset in the main lines. As a product manager of was navigating her data, she said that she would be curious to see a list of all the paths, and I realised that a summary of the paths this could be a valuable overview for data producers.

In *Chapter 4* PATH OUTLINES, I presented a tool to support RDF data producers in browsing path-based summaries of their datasets. I made the hypothesis that current RDF summary approaches were limited by their granularity and that the path could be a meaningful granularity to summarise Knowledge Graphs for data producers. I refined the concept of *semantic path* and renamed it *path outline* to better convey the idea of summary. I designed an interface based on coordinated views with 2 novel visualisations, which allow to represent a very large number of *path outlines*, browse through them by meaningful chunks and inspect their metrics. Thanks to the collaboration with

Wimmics team, the tool can also summarise chains of triples continuing to another dataset. I conducted a controlled study comparing *Path Outlines* with the current baseline technique (Virtuoso SPARQL query editor) in an experiment with 36 participants. Participants preferred *Path Outlines*, found it easier and more comfortable to use, were faster and had better task completion and fewer errors with it.

A limitation of *Path Outlines*, however, was that the statistics are computed once for the whole dataset, for each path separately. This makes it possible to generate a list of entities missing a specific path, but those entities then have to be inspected one after another, and this for each path. Therefore, in *Chapter 5* THE MISSING PATH, I investigated the use of multidimensional vectors to identify subsets of items missing the same paths. My hypothesis was that identifying groups of entities sharing the same structure could help detect causes, and allow to fix the entities as groups, saving significant time. I designed an interface combining a map with clusters of entities missing the same paths, together with statistical summaries of paths for the collection, that can be compared with statistical summaries of paths for any subset. The summaries show the profile of paths both in terms of completeness and distribution of values. I related the iterative design process and the evaluation of the tool with Wikidata contributors, who deal with particularly heterogeneous data. Participants gained new insights on incompleteness in their data, using various exploratory strategies supported by the coordination between the map and the summaries.

Those 3 applications of the concept demonstrate the potential of using the path as the granularity to produce automatic and meaningful browsable overviews of Knowledge Graphs.

6.2 HIGH-LEVEL INSIGHTS

6.2.1 *Information space*

Those tools go from overview to detail in different ways, letting users access different “aspects of the information space” [54].

S-Paths is a generic application to browse the *content* of the data. Its strategy to avoid information overload is to select *cuts* at different scales, based on statistics, combined with heuristics defining a readable and efficient visualisation. The selection of paths based on their readability and completeness produces interpretable associations, possibly unexpected. The overviews are selective; they show only a few of the many facets of a collection. Users gain *insights* as they progress through the successive snapshots with advanced

transitions to provide continuity. The browsing is exploratory and opportunistic. It is meant to raise questions, to catch users' attention and call for further exploration, so that they feel like engaging with unknown data before they know if and how they could be interested in interacting with them.

Path Outlines is also generic and oriented towards discovery, but this time by browsing the *structure* of the data. Considering the structure as the content to browse shifts the problem; the dimensions to filter by are now related to the structure; the combinatory mechanism can be untangled and laid flat in relatively even columns with a manageable number of elements. The overviews of the collections are meant to be exhaustive, as data producers need to control every statement. I think the tool was successful in providing users with a simple and usable conceptual model of high-level information in a Knowledge Graph [83]. Most participants, with various levels of expertise and various backgrounds, were able to use it easily.

Focused on the completeness, *The Missing Path* lets users browse *summaries* of the content that become more precise as the scope narrows. The summary of the collection acts as an anchor, a referent and immutable context, against which users can compare the summaries of the subset. This new kind of browsing—that we could call *comparative browsing*, is meant to support analytic tasks. The evaluation helped us solve major usability problems and clarify the different mechanisms involved in this browsing process. However, the mechanisms are still new, and I think that it will take time to refine the principles and make the exploration feel easy. The goal for the next step is to make users feel comfortable with the various levels of aggregation involved in a Knowledge Graph.

6.2.2 *Data processing*

The possibility of interactive exploration is very dependant on response time. The activity of bowsing implies that it is possible to look at items one after another, and this can only happen if detailed information is available relatively rapidly. When an application takes too long to respond, users do not try to explore items unless they are sure they correspond to a very precise need.

Conceptually, the design of *S-Paths* should scale, but technically, its implementation, directly plugged to the SPARQL endpoint, does only to a certain extent . The crux of the problem is that queries returning no results can take a very long time to execute, and the response time increases with the size of a dataset. Therefore, checking if a combination of properties is valid can have a high cost. I had underestimated this issue, thinking that statistically, thanks to

the pre-analysis of paths and the flexibility of view requirements, the system would not have to check many combinations before finding a satisfying one. But it turned out that even those few queries could take very long, and put the server under heavy load. Even more annoying is the fact that it tends to happen on small subsets, which is difficult to understand for users, who expect the waiting time to be correlated with the amount of data to process.

Furthermore, I encountered another problem, which I think is interesting to mention, although I don't have precise measures to describe it. I noticed that after intensive use over weeks, *S-Paths* would slow down. I identified that some of the queries would return no results when they should have. Reinstalling the virtuoso database would resolve the issue. It looked like the types of queries sent by *S-Paths* (set-based queries over long paths) might create confusion in indexes.

The implementation of *The Missing Path*, extracting a matrix of values, with a column per paths and a row per entity, proved much more efficient. It was then quick and easy to reduce the matrix, for any subset of entities and any subset of paths, before processing it. The response time was much more reliable and predictable, correlated to the size of the subset queried. I was reluctant to use this approach at first because it implies to extract the data, and therefore to manage updates. But I now think this is a necessary step to provide advanced interaction on RDF data at various scales. The good news is that it prepares the data in a shape that enables machine learning processing, and thus opens perspectives for advanced exploratory analysis.

6.3 OPEN PERSPECTIVES

6.3.1 *Path outlines as interoperable metadata*

Path outlines provide useful metadata about a dataset. We saw how they could support 3 different applications, and I think they could be reused by many others. The creation of an ontology for their description—or an extension to an existing ontology like, for instance, the VOID vocabulary [5]—would enable data producers to publish them in an interoperable format along with their dataset. Those metadata could be beneficial indicators for the interpretation and reuse of a dataset; they would serve as a pointer to the collections of interest, and the depth until which it is relevant to explore them.

Data reusers could then browse any dataset with tools such as *Path Outlines*, with no further analysis. For exploratory tools like *The Missing Path*, which require additional processing on the fly to analyse subsets in a collec-

tion, they would still be helpful to bootstrap the system, indicating the paths of interest and providing the initial visualisation.

6.3.2 *Visualising the structure of Knowledge Graphs*

As mentioned, a limitation of our tool *Path Outlines* is to group paths by depth, keeping users from seeing at the same time paths of various depths. In addition to slowing down tasks involving paths of various depths, this prevents the representation of full 'sentence graphs' [140]. Sentence graphs are higher-level statements similar to paths, with the difference that they can involve several branches in the graph. Back to Fig. 1, a sentence graph could be: 'Marie Curie is a female Person born in 1867 and affiliated to la Sorbonne University in Paris'. Such structures are already used to summarise the most frequent patterns in a graph [13], but they are meant to be processed by machines. Presenting a series of node-link diagrams to humans seems complicated, given the difficulties we have to read a single one.

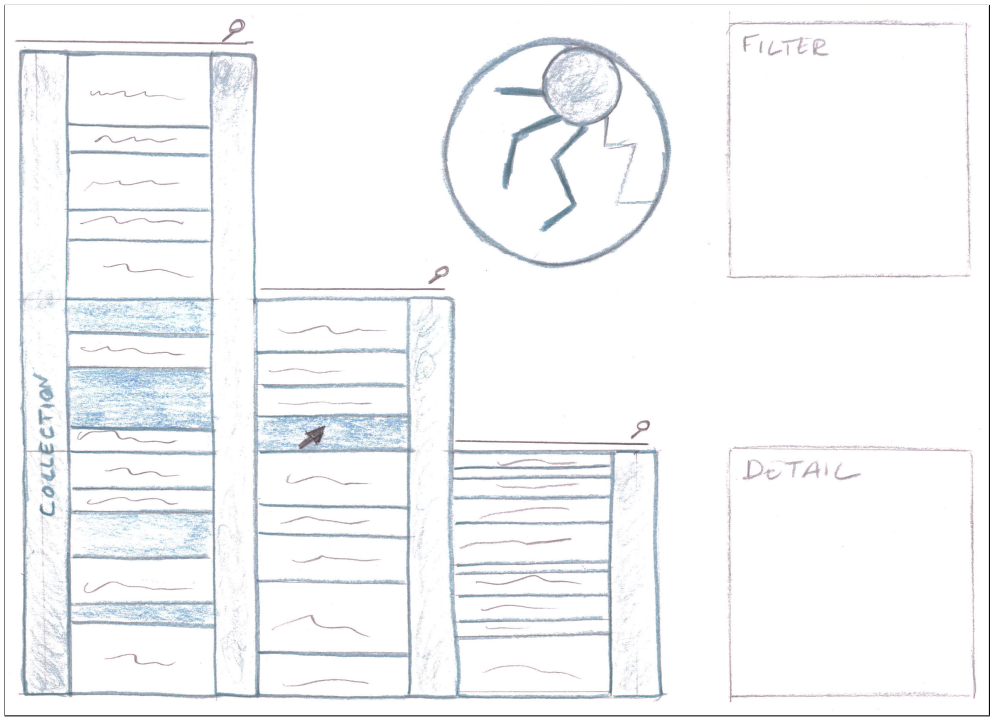


Figure 59: Sketch of a new version of the *path browser*, supporting simultaneously paths of various depths.

The current version of *Path Outlines* already allows to read sentence graphs; however its limit to a specific depth at a time means that users can either read 'Marie Curie is a female Person born in 1867' (depth 1) or 'Marie Curie is affiliated to la Sorbonne University in Paris' (depth 2), and not the full sentence. If the *path browser* could support different depths (Fig. 59), it could be used to

display such summaries of sentence graphs to humans. One of the applications of their analysis being to optimise indexes in a database, this visualisation could be useful to monitor such indexes. It could also help programmers optimise their queries.

In another context, this new *path browser* could also support ontologists in evaluating the impact of an edit in the model. Indeed, a problem they have when they design an ontology is that they cannot see the influence of a change made to a specific element on other elements [128]. Seeing the impact on the statements produced could help them refine their design while ensuring backward compatibility.

Technically speaking, I think that *path outlines* would still be the most efficient abstraction to write algorithms displaying such summary graphs, computing the metrics for associations on top. At least, I can precisely imagine how to program such an application, while it is not clear to me how to handle sentence graphs as a basic unit.

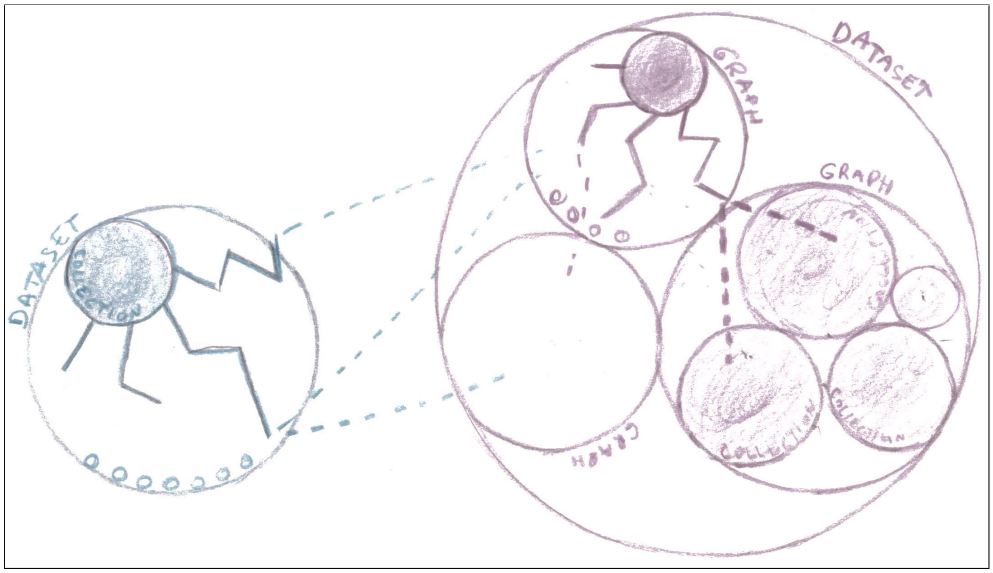


Figure 60: Sketch of an extended version of the *broken outlines* visualisation, showing links between the different datasets and their named graphs.

The *broken outlines* visualisation could also be developed to better show the flows of interlinks between datasets, as in Fig. 60. If *paths outlines* were published as metadata, this could provide a valuable visualisation for the LOD cloud (lod-cloud.net). Just as in *Path Outlines*, datasets which expose metadata could be opened and inspected, while others would be able to receive links declared by others and display basic information such as their size and name on hover.

I also have the intuition that *path outlines* could be used in interlinking applications, although I don't have any precise visualisation in mind yet. As of today, data producers, when they interlink their data with another dataset, have two massive lists of entities, each on one side of the screen, and must indicate equivalence links between entities. This activity is very difficult when not seeing the links in context. They often need to open entities in new tabs to see their full description, which is very impractical and time-consuming. A visualisation based on *path outlines* descriptions could probably support the display and comparison of entities in context, to let users better evaluate the nature and degree of similarity.

6.3.3 *Exploring the content of Knowledge Graphs*

Other types of applications might also beneficiate from the fact that *path outlines* provide a unidimensional space for descriptions of various depths. We used it to support visual exploratory analysis of completeness with *The Missing Path*, and we are already working on applying it to the analysis of other quality dimensions. With *S-Paths* as with *The Missing Path*, *path outlines* offer a visual space that is relatively 'cheap' to compute, for users to spot correlations visually, saving the cost of analysing an uncountable number of combinations. I think that this information space, combined with simple heuristics as those used in *S-Paths* (most complete paths, datatypes, number of unique value) can offer a bridge to multivariate data visualisation [69] and graph visualisation. For instance, I could imagine automatically selecting the most represented properties offering a continuum or having a low number of unique values, and using their path outlines summaries to configure a parallel coordinates visualisation. Updates could be computed on the fly through progressive rendering based on WebSockets.

Besides visualisation, *path outlines* could also support applications to explore the content of Knowledge Graphs programmatically, especially when federated queries are involved. Indeed, the cost of executing queries over collections along paths, and across federated endpoints is quite high, especially when it returns no results. It would be very useful to know in advance which queries are likely to return significant results.

Finally, Knowledge Graph embedding techniques similar to the one we developed in *The Missing Path* are used to support machine learning over RDF, with applications as diverse analysing the contents of datasets [139], performing learning [56], estimating the similarity of items [4] or supporting rec-

ommendation [45, 79, 87]. *path outlines* descriptors could provide a standard way to generate and manage the embeddings.

6.3.4 *Evaluating understanding*

The tools I develop have in common to be designed to support understanding. Therefore, their efficiency is difficult to evaluate, and I struggled to design evaluations which I am not fully satisfied with. After taking a step back, I think a possible evaluation for *S-Paths* would be to compare it with other types of tools, such as a paginated list faceted browser and a visualisation system (as described in Sect. 2.3.2), giving participants 20 minutes to write a summary of the content. We could measure the number of errors, the number of correct statements and their level (using Bertin's levels mentioned in Sect. 3.1). Such an evaluation would be valuable to gain insights about the different types of understanding provided by different types of overviews and navigation. I find it important to invest efforts in designing evaluations oriented towards open cognition tasks because Knowledge Graphs are about knowledge.

6.4 THE END... IS A NEW BEGINNING

Besides visualising RDF datasets, my interest in pursuing this research was to learn how to use HCI and visualisation techniques to address problems users have when they need to access complex information. I wanted to learn how to situate, refine and evaluate an idea. I have the feeling I am just beginning to master elements of method, and I would like to start this thesis all over again. I guess I just have to accept that it could have been better. Luckily, there is still (half!) a life waiting for me to experiment those new skills.

BIBLIOGRAPHY

[1] Z. Abedjan et al. “Profiling and mining RDF data with ProLOD++”. In: *2014 IEEE 30th International Conference on Data Engineering (ICDE)*. Los Alamitos, CA, USA: IEEE Computer Society, 2014, pp. 1198–1201. DOI: 10.1109/ICDE.2014.6816740.

[2] James Abello, Frank Van Ham, and Neeraj Krishnan. “Ask-graphview: A large scale graph visualization system”. In: *IEEE transactions on visualization and computer graphics* 12.5 (2006), pp. 669–676. DOI: 10.1109/TVCG.2006.120.

[3] Manel Achichi et al. “DOREMUS: a graph of linked musical works”. In: *The semantic web – ISWC 2018*. Ed. by Denny Vrandečić et al. Springer International Publishing, 2018, pp. 3–19. DOI: 10.1007/978-3-030-00671-6_1.

[4] Hogan Aidan et al. “Some entities are more equal than others: statistical methods to consolidate Linked Data”. In: *Workshop on New Forms of Reasoning for the Semantic Web: Scalable & Dynamic*. Ed. by Stefano Ceri et al. Proceedings of the Workshop on New Forms of Reasoning for the Semantic Web: Scalable & Dynamic. Heraklion, Greece, May 2010. URL: https://hal.archives-ouvertes.fr/hal-01240427.

[5] Keith Alexander et al. *Describing Linked Datasets with the VoID Vocabulary*. 2011. URL: https://www.w3.org/TR/void/.

[6] Marije van Amelsvoort et al. “The importance of design in learning from node-link diagrams”. In: *Instructional science* 41.5 (2013), pp. 833–847. DOI: 10.1007/s11251-012-9258-x.

[7] SFCD Araújo, Daniel Schwabe, and Simone Barbosa. “Experimenting with explorator: a direct manipulation generic rdf browser and querying tool”. In: 443 (2009). Ed. by Siegfried Handschuh, Tom Heath, and VinhTuan Thai. URL: http://ceur-ws.org/Vol-443/.

[8] Daniel Archambault, Tamara Munzner, and David Auber. “Tugging graphs faster: Efficiently modifying path-preserving hierarchies for browsing paths”. In: *IEEE Transactions on Visualization and Computer Graphics* 17.3 (2010), pp. 276–289. DOI: 10.1109/TVCG.2010.60.

[9] Marcelo Arenas et al. “SemFacet: Semantic faceted search over yago”. In: *Proceedings of the 23rd international conference on world wide web*. WWW ’14 companion. Association for Computing Machinery, 2014, 123–126. DOI: 10.1145/2567948.2577011.

[10] Ghislain Auguste Atemezing and Raphaël Troncy. “Towards a Linked-Data based Visualization Wizard.” In: *COLD 2014 – Consuming Linked Data – Proceedings of the 5th International Workshop on Consuming Linked Data (COLD 2014) co-located with the 13th International Semantic Web Conference (ISWC 2014) Riva del Garda, Italy, October 20, 2014*. Ed. by Olaf Hartig, Aidan Hogan, and Juan Sequeda. 2014. URL: http://ceur-ws.org/Vol-1264/cold2014_AtemezingT.pdf.

[11] Sören Auer et al. “LODStats – An Extensible Framework for High-Performance Dataset Analytics”. In: *Knowledge Engineering and Knowledge Management*. Ed. by Annette ten Teije et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 353–362. DOI: 10.1007/978-3-642-33876-2_31.

[12] Benjamin Bach et al. “OntoTrix: a hybrid visualization for populated ontologies”. In: *Proceedings of the 20th international conference companion on World wide web*. 2011, pp. 177–180. DOI: 10.1145/1963192.1963283.

[13] Adrien Basse et al. “DFS-based frequent graph pattern extraction to characterize the content of RDF Triple Stores”. In: *Web Science Conference 2010 (WebSci10)*. Raleigh, United States, Apr. 2010. URL: https://hal.inria.fr/hal-01170896.

[14] Mohamed Ben Ellefi et al. “RDF dataset profiling–a survey of features, methods, vocabularies and applications”. In: *Semantic Web* 9.5 (2018), 677–705. DOI: 10.3233/SW-180294.

[15] T. Berners-Lee et al. “Tabulator: Exploring and analyzing linked data on the semantic web”. In: *Proceedings of the 3rd international semantic web user interaction workshop*. Ed. by Christian Bizer et al. Vol. 2006. 2006, p. 159. URL: http://ceur-ws.org/Vol-369/.

[16] J. Bertin. *La graphique et le traitement graphique de l’information*. Nouvelle Bibliothèque Scientifique. Flammarion, 1977. ISBN: 978-2-08-211112-6.

[17] Christian Bizer and Richard Cyganiak. “Quality-driven information filtering using the WIQA policy framework”. In: *Journal of Web Semantics* 7.1 (2009). The Semantic Web and Policy, pp. 1 –10. ISSN: 1570-8268. DOI: 10.1016/j.websem.2008.02.005.

[18] Dan Brickley and Libby Miller. *FOAF Vocabulary Specification 0.99*. Ontology. Feb. 2014. URL: http://xmlns.com/foaf/spec/.

[19] Sören Brunk and Philipp Heim. “tFacet: Hierarchical faceted exploration of semantic data using well-known interaction concepts”. In: *Proceedings of the International Workshop on Data-Centric Interactions on the Web in conjunction with the 13th IFIP TC13 Conference on Human-Computer-Interaction (INTERACT 2011) Lisbon, Portugal, Sep 6, 2011*. Ed. by Paloma Díaz et al. 2011, 31–36. URL: http://ceur-ws.org/Vol-817/paper3.pdf.

[20] C. Böhm et al. “Profiling linked open data with ProLOD”. In: *2010 IEEE 26th international conference on data engineering workshops (ICDEW 2010)*. Citation Key: 5452762. 2010, 175–178. DOI: 10.1109/ICDEW.2010.5452762.

[21] Diego Valerio Camarda, Silvia Mazzini, and Alessandro Antonuccio. “LodLive, Exploring the Web of Data”. In: *I-SEMANTICS ’12: Proceedings of the 8th International Conference on Semantic Systems*. Ed. by Harald Sack and Tassilo Pellegrini. New York, NY, USA: ACM, 2012, pp. 197–200. DOI: 10.1145/2362499.2362532.

[22] Jeremy Carroll and Graham Klyne. *Resource Description Framework (RDF): Concepts and Abstract Syntax*. W3C Recommendation. W3C, Feb. 2004. URL: http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/.

[23] James Clark, Steve DeRose, et al. *XML path language (XPath)*. 1999.

[24] Edward C. Clarkson, Shamkant B. Navathe, and James D. Foley. “Generalized Formal Models for Faceted User Interfaces”. In: *Proceedings of the 9th ACM/IEEE-CS Joint Conference on Digital Libraries*. JCDL ’09. Austin, TX, USA: Association for Computing Machinery, 2009, 125–134. ISBN: 9781605583228. DOI: 10.1145/1555400.1555422.

[25] Charles R. Collins and Kenneth Stephenson. “A circle packing algorithm”. In: *Computational Geometry* 25.3 (2003), pp. 233 –256. ISSN: 0925-7721. DOI: 10.1016/S0925-7721(02)00099-8.

[26] Marie Destandau and Jean-Daniel Fekete. *The Missing Path: Diagnosing Incompleteness in Linked Data*. 2020. arXiv: 2005.08101 [cs.HC].

[27] Marie Destandau and Jean-Daniel Fekete. "The Missing Path: Diagnosing Incompleteness in Linked Data". working paper or preprint. May 2020. URL: https://hal.archives-ouvertes.fr/hal-02612896.

[28] Marie Destandau et al. "Path Outlines: Browsing Path-Based Summaries of Linked Open Datasets". In: *ArXiv e-prints* (2020). arXiv: 2002.09949 [cs.HC].

[29] Marie Destandau, Caroline Appert, and Emmanuel Pietriga. "S-Paths: Set-Based Visual Exploration of Linked Data Driven by Semantic Paths". In: *Semantic Web* (2020). DOI: 10.3233/SW-200383.

[30] Diana DeStefano and Jo-Anne LeFevre. "Cognitive load in hypertext reading: A review". In: *Computers in human behavior* 23.3 (2007). DOI: 10.1016/j.chb.2005.08.012.

[31] Marek Dudáš and Vojtech Svátek. "Discovering Issues in Datasets Using LODSight Visual Summaries". In: *Proceedings of the International Workshop on Visualizations and User Interfaces for*. 2015, p. 77. URL: http://ceur-ws.org/Vol-1456/.

[32] Marek Dudáš, Vojtěch Svátek, and Jindřich Mynarz. "Dataset summary visualization with lodsight". In: *European Semantic Web Conference*. Springer. 2015, pp. 36–40. DOI: 10.1007/978-3-319-25639-9_7.

[33] Basil Ell, Denny Vrandečić, and Elena Simperl. "Labels in the Web of Data". In: *The Semantic Web – ISWC 2011*. Ed. by Lora Aroyo et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 162–176. ISBN: 978-3-642-25073-6. DOI: 10.1007/978-3-642-25073-6_11.

[34] Stef Van den Elzen and Jarke J Van Wijk. "Multivariate network exploration and presentation: From detail to overview via selections and aggregations". In: *IEEE Transactions on Visualization and Computer Graphics* 20.12 (2014), pp. 2310–2319. DOI: 10.1109/TVCG.2014.2346441.

[35] Ivan Ermilov et al. "Linked open data statistics: Collection and exploitation". In: *International Conference on Knowledge Engineering and the Semantic Web*. Springer. 2013, pp. 242–249. DOI: 10.1007/978-3-642-41360-5_19.

[36] Fredo Erxleben et al. “Introducing Wikidata to the linked data web”. In: *International Semantic Web Conference*. Springer. 2014, pp. 50–65. DOI: 10.1007/978-3-319-11964-9_4.

[37] Sébastien Ferré. “Conceptual Navigation in RDF Graphs with SPARQL-Like Queries”. In: *Formal Concept Analysis*. Ed. by Léonard Kwuida and Barış Sertkaya. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 193–208. ISBN: 978-3-642-11928-6. DOI: 10.1007/978-3-642-11928-6_14.

[38] Sébastien Ferré. “Sparklis: an expressive query builder for SPARQL endpoints with guidance in natural language”. In: *Semantic Web* 8.3 (2017), pp. 405–418. DOI: 10.3233/SW-150208.

[39] J. Fuchs et al. “A Systematic Review of Experimental Studies on Data Glyphs”. In: *IEEE Transactions on Visualization and Computer Graphics* 23.7 (2017), pp. 1863–1879. DOI: 10.1109/TVCG.2016.2549018.

[40] Luis Fuenmayor et al. “FaRBIE: A faceted reactive browsing interface for multi RDF knowledge graph exploration”. In: *VOILA 2017 Visualization and Interaction for Ontologies and Linked Data - Proceedings of the Third International Workshop on Visualization and Interaction for Ontologies and Linked Data - co-located with the 16th International Semantic Web Conference (ISWC 2017) - Vienna, Austria, October 22, 2017.* Ed. by Valentina Ivanova et al. Vol. 1947. 2017, 111–122. URL: http://ceur-ws.org/Vol-1947/.

[41] Roberto García and Rosa Gil. “Improving human-semantic web interaction: The rhizomer experience”. In: *SWAP 2006 Semantic Web Applications and Perspectives - Proceedings of the 3rd Italian Semantic Web Workshop - Scuola Normale Superiore, Pisa, Italy, 18-20 December, 2006*. Ed. by Giovanni Tummarello, Paolo Bouquet, and Oreste Signore. Vol. 201. 2006. URL: http://ceur-ws.org/Vol-201/.

[42] *Geonames*. https://www.geonames.org/ontology/documentation.html. Accessed January 06, 2021.

[43] Mohammad Ghoniem, J-D Fekete, and Philippe Castagliola. “A comparison of the readability of graphs using node-link and matrix-based representations”. In: *IEEE Symposium on Information Visualization*. Ieee. 2004, pp. 17–24. DOI: 10.1109/INFVIS.2004.1.

[44] Michael Gleicher et al. “Visual comparison for information visualization”. In: *Information Visualization* 10.4 (2011), pp. 289–309. DOI: 10.1177/1473871611416549.

[45] Sébastien Harispe et al. “Semantic Measures Based on RDF Projections: Application to Content-Based Recommendation Systems”. In: *On the Move to Meaningful Internet Systems: OTM 2013 Conferences*. Sept. 2013, pp. 606–615. ISBN: 9783642410291. DOI: 10.1007/978-3-642-41030-7_44.

[46] Andreas Harth and Sebastian Speiser. “On Completeness Classes for Query Evaluation on Linked Data”. In: *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*. AAAI’12. Toronto, Ontario, Canada: AAAI Press, 2012, 613–619. DOI: 10.5555/2900728.2900816.

[47] J. Heer and G. Robertson. “Animated Transitions in Statistical Data Graphics”. In: *IEEE Transactions on Visualization and Computer Graphics* 13.6 (Nov. 2007), pp. 1240–1247. DOI: 10.1109/TVCG.2007.70539.

[48] Philipp Heim, Jürgen Ziegler, and Steffen Lohmann. “gFacet: A browser for the web of data”. In: *IMC-SSW’08 Interacting with Multimedia Content in the Social Semantic Web - Proceedings of the International Workshop on Interacting with Multimedia Content in the Social Semantic Web (IMC-SSW’08) - Koblenz, Germany, December 3, 2008*. Ed. by Sören Auer et al. Vol. 417. 2008, 49–58. URL: http://ceur-ws.org/Vol-417/.

[49] Philipp Heim et al. “RelFinder: Revealing Relationships in RDF Knowledge Bases”. In: *Semantic Multimedia – 4th International Conference on Semantic and Digital Media Technologies, SAMT 2009 Graz, Austria, December 2-4, 2009 Proceedings*. Ed. by Tat-Seng Chua et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 182–187. DOI: 10.1007/978-3-642-10543-2.

[50] Philipp Heim et al. “SemLens: Visual Analysis of Semantic Data with Scatter Plots and Semantic Lenses”. In: *Proceedings of the 7th International Conference on Semantic Systems*. I-Semantics ’11. Graz, Austria: ACM, 2011, pp. 175–178. ISBN: 978-1-4503-0621-8. DOI: 10.1145/2063518.2063543.

[51] Ivan Herman, Guy Melançon, and M Scott Marshall. “Graph visualization and navigation in information visualization: A survey”. In: *IEEE Transactions on visualization and computer graphics* 6.1 (2000), pp. 24–43. DOI: 10.1109/2945.841119.

[52] M. Hildebrand, J. van Ossenbruggen, and L. Hardman. "/facet: A Browser for Heterogeneous Semantic Web Repositories". In: *The Semantic Web - ISWC 2006 - 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5-9, 2006. Proceedings*. Ed. by I. Cruz et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 272–285. DOI: 10.1007/11926078.

[53] Aidan Hogan et al. "Knowledge graphs". In: *arXiv preprint arXiv:2003.02320* (2020).

[54] Kasper Hornbæk and Morten Hertzum. "The notion of overview in information visualization". In: *International Journal of Human-Computer Studies* 69.7 (2011), pp. 509 –525. ISSN: 1071-5819. DOI: https://doi.org/10.1016/j.ijhcs.2011.02.007.

[55] Weidong Huang and Peter Eades. "How people read graphs". In: *proceedings of the 2005 Asia-Pacific symposium on Information visualisation-Volume 45*. Australian Computer Society, Inc. 2005, pp. 51–58. DOI: 10.5555/1082315.1082324.

[56] Yi Huang et al. "A scalable approach for statistical learning in semantic graphs". In: *Semantic Web* 5.1 (2014), pp. 5–22. DOI: 10.3233/SW-130100.

[57] Thomas Hubauer et al. "Use Cases of the Industrial Knowledge Graph at Siemens". In: *Proceedings of the ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks, co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, USA, October 8th to 12th, 2018*. Ed. by Marieke van Erp et al. Vol. 2180. URL: http://ceur-ws.org/Vol-2180/.

[58] David Huynh, Stefano Mazzocchi, and David Karger. "Piggy bank: Experience the semantic web inside your web browser". In: *The semantic web – ISWC 2005*. Ed. by Yolanda Gil et al. Springer Berlin Heidelberg, 2005, 413–430. DOI: 10.1007/11574620_31.

[59] David F. Huynh and David R. Karger. *Parallax and Companion: Set-based Browsing for the Data Web*. 2009. URL: http://davidhuynh.net/media/papers/2009/www2009-parallax.pdf.

[60] Eero Hyvönen. "Using the Semantic Web in Digital Humanities: Shift from data publishing to data-analysis and serendipitous knowledge discovery". In: *Semantic Web* Preprint (2020), pp. 1–7.

[61] Subhi Issa et al. “Revealing the Conceptual Schemas of RDF Datasets”. In: *International Conference on Advanced Information Systems Engineering*. Springer. 2019, pp. 312–327. DOI: 10.1007/978-3-030-21290-2_20.

[62] Sean Kandel et al. “Profiler: Integrated statistical analysis and visualization for data quality assessment”. In: *Proceedings of the International Working Conference on Advanced Visual Interfaces*. 2012, pp. 547–554. DOI: 10.1145/2254556.2254659.

[63] Ali Khalili, Peter van den Besselaar, and Klaas Andries de Graaf. “FERASAT: A Serendipity-Fostering Faceted Browser for Linked Data”. In: *Proceedings of the European Semantic Web Conference*. ESWC ’18. Springer, 2018, pp. 351–366. DOI: 10.1007/978-3-319-93417-4_23.

[64] Georgi Kobilarov and Ian Dickinson. “Humboldt: Exploring linked data”. In: 369 (2008). Ed. by Christian Bizer et al. URL: http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-369/paper15.pdf.

[65] Jörg Koch and Thomas Franz. “LENA - Browsing RDF Data More Complex Than Foaf”. In: *The Semantic Web – ISWC 2008 – 7th International Semantic Web Conference, ISWC 2008, Karlsruhe, Germany, October 26-30, 2008, Proceedings – Posters and Demonstrations*. Ed. by Amit P. Sheth et al. 2008. DOI: 10.1007/978-3-540-88564-1.

[66] T. von Landesberger et al. “Visual Analysis of Large Graphs: State-of-the-Art and Future Research Challenges”. In: *Computer Graphics Forum* 30.6 (2011), pp. 1719–1749. DOI: 10.1111/j.1467-8659.2011.01898.x.

[67] Danh Le-Phuoc et al. “The Graph of Things: A step towards the Live Knowledge Graph of connected things”. In: *Journal of Web Semantics* 37 (2016), pp. 25–35. DOI: 10.1016/j.websem.2016.02.003.

[68] Bongshin Lee et al. “Treeplus: Interactive exploration of networks with enhanced tree layouts”. In: *IEEE Transactions on Visualization and Computer Graphics* 12.6 (2006), pp. 1414–1426. DOI: 10.1109/TVCG.2006.106.

[69] S. Liu et al. “Visualizing High-Dimensional Data: Advances in the Past Decade”. In: *IEEE Transactions on Visualization and Computer Graphics* 23.3 (2017), pp. 1249–1268. DOI: 10.1109/TVCG.2016.2640960.

[70] G. Marchionini and B. Shneiderman. “Finding facts vs. browsing knowledge in hypertext systems”. In: *Computer* 21.1 (1988), pp. 70–80. DOI: 10.1109/2.222119.

[71] Nicolas Marie and Fabien Gandon. “Survey of linked data based exploration systems”. In: *IESD 2014 – Intelligent Exploration of Semantic Data – Proceedings of the 3rd International Workshop on Intelligent Exploration of Semantic Data (IESD 2014) co-located with the 13th International Semantic Web Conference (ISWC 2014) Riva del Garda, Italy, October 20, 2014.* Oct. 2014. URL: https://hal.inria.fr/hal-01057035.

[72] Nicolas Marie, Fabien Gandon, and Myriam Ribière. “Exploratory Search on the Top of DBpedia Chapters with the Discovery Hub Application”. In: *The Semantic Web: ESWC 2013 Satellite Events*. Ed. by Philipp Cimiano et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 287–288. DOI: 10.1007/978-3-642-41242-4_45.

[73] Fintan McGee et al. “The state of the art in multilayer network visualization”. In: *Computer Graphics Forum*. Vol. 38. 6. Wiley Online Library. 2019, pp. 125–149. DOI: 10.1111/cgf.13610.

[74] L. McInnes, J. Healy, and J. Melville. “UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction”. In: *ArXiv e-prints* abs/1802.03426 (Feb. 2018). arXiv: 1802.03426 [stat.ML].

[75] Pablo N Mendes, Hannes Mühleisen, and Christian Bizer. “Sieve: linked data quality assessment and fusion”. In: *Proceedings of the 2012 Joint EDBT/ICDT Workshops*. Citeseer. 2012, pp. 116–123. DOI: 10.1145/2320765.2320803.

[76] Nandana Mihindukulasooriya et al. “Loupe-An Online Tool for Inspecting Datasets in the Linked Data Cloud.” In: *Proceedings of the ISWC 2015 Posters & Demonstrations Track co-located with the 14th International Semantic Web Conference (ISWC-2015) Bethlehem, PA, USA, October 11, 2015.* Ed. by Serena Villata, Jeff Z. Pan, and Mauro Dragoni. 2015. URL: http://ceur-ws.org/Vol-1486/.

[77] Matthew Miller, Jeff Walloch, and M Cristina Pattuelli. “Visualizing linked Jazz: A web-based tool for social network analysis and exploration”. In: *Proceedings of the American Society for Information Science and Technology* 49.1 (2012), pp. 1–3. DOI: 10.1002/meet.14504901295.

[78] Claudia Müller-Birn et al. “Peer-Production System or Collaborative Ontology Engineering Effort: What is Wikidata?” In: *Proceedings of the 11th International Symposium on Open Collaboration*. OpenSym ’15. San Francisco, California: Association for Computing Machinery, 2015, pp. 1–10. ISBN: 9781450336666. DOI: 10.1145/2788993.2789836. URL: https://doi.org/10.1145/2788993.2789836.

[79] Cataldo Musto. “Enhanced Vector Space Models for Content-Based Recommender Systems”. In: *Proceedings of the Fourth ACM Conference on Recommender Systems*. RecSys ’10. New York, NY, USA: Association for Computing Machinery, 2010, 361–364. ISBN: 9781605589060. DOI: 10.1145/1864708.1864791. URL: https://doi.org/10.1145/1864708.1864791.

[80] Eetu Mäkelä. “Aether – generating and viewing extended VoID statistical descriptions of RDF datasets”. In: *The semantic web: ESWC 2014 satellite events*. Ed. by Valentina Presutti et al. Springer International Publishing, 2014, 429–433. ISBN: 978-3-319-11955-7. DOI: 10.1007/978-3-319-11955-7_61.

[81] Carolina Nobre et al. “The State of the Art in Visualizing Multivariate Networks”. In: *Computer Graphics Forum (EuroVis)* 38 (2019), pp. 807–832. DOI: 10.1111/cgf.13728.

[82] Luis Gustavo Nonato and Michael Aupetit. “Multidimensional Projection for Visual Analytics: Linking Techniques with Distortions, Tasks, and Layout Enrichment”. In: *IEEE Trans. Vis. Comput. Graphics* 25.8 (2018), pp. 2650–2673. ISSN: 2160-9306. DOI: 10.1109/TVCG.2018.2846735.

[83] Donald A Norman. *Living with complexity*. MIT press, 2016. ISBN: 9780262528948. URL: https://mitpress.mit.edu/books/living-complexity.

[84] Laura R Novick. “Understanding spatial diagram structure: An analysis of hierarchies, matrices, and networks”. In: *The Quarterly Journal of Experimental Psychology* 59.10 (2006). the hierarchy depicts a rigid structure of power or precedence relations among items, pp. 1826–1856. DOI: 10.1080/17470210500298997.

[85] Laura R Novick and Sean M Hurley. “To matrix, network, or hierarchy: That is the question”. In: *Cognitive psychology* 42.2 (2001), pp. 158–216. DOI: 10.1006/cogp.2000.0746.

[86] Eyal Oren, Renaud Delbru, and Stefan Decker. “Extending faceted navigation for RDF data”. In: *The semantic web - ISWC 2006*. Ed. by Isabel Cruz et al. Springer Berlin Heidelberg, 2006, 559–572. DOI: 10.1007/11926078_40.

[87] Enrico Palumbo et al. “Knowledge Graph Embeddings with node2vec for Item Recommendation”. In: *The Semantic Web: ESWC 2018 Satellite Events*. Ed. by Aldo Gangemi et al. Cham: Springer International Publishing, 2018, pp. 117–120. ISBN: 978-3-319-98192-5. DOI: 10.1007/978-3-319-98192-5_22.

[88] Jeff Z Pan et al. *Exploiting linked data and knowledge graphs in large organisations*. Springer, 2017. DOI: 10.1007/978-3-319-45654-6.

[89] Christian Partl et al. “Pathfinder: Visual analysis of paths in graphs”. In: *Computer Graphics Forum*. Vol. 35. 3. Wiley Online Library. 2016, pp. 71–80. DOI: 10.1111/cgf.12883.

[90] Adam Perer and Ben Shneiderman. “Integrating statistics and visualization: case studies of gaining clarity during exploratory data analysis”. In: *Proceedings of the SIGCHI conference on Human Factors in computing systems*. ACM. 2008, pp. 265–274. DOI: 10.1145/1357054.1357101.

[91] Adam Perer and Ben Shneiderman. “Systematic yet Flexible Discovery: Guiding Domain Experts through Exploratory Data Analysis”. In: *Proceedings of the 13th International Conference on Intelligent User Interfaces*. IUI ’08. Gran Canaria, Spain: Association for Computing Machinery, 2008, 109–118. ISBN: 9781595939876. DOI: 10.1145/1378773.1378788.

[92] D. Petrelli et al. “Multi Visualization and Dynamic Query for Effective Exploration of Semantic Data”. In: *The semantic web – ISWC 2009 – Proceedings of the 8th International Semantic Web Conference*. Ed. by A. Bernstein et al. ISWC’09. Springer Berlin Heidelberg, 2009, 505–520. DOI: 10.1007/978-3-642-04930-9_32.

[93] Vsevolod Peysakhovich, Christophe Hurter, and Alexandru Telea. “Attribute-driven edge bundling for general graphs with applications in trail analysis”. In: *2015 IEEE Pacific Visualization Symposium (PacificVis)*. IEEE. 2015, pp. 39–46.

[94] Robert Pienta et al. “Scalable graph exploration and visualization: Sensemaking challenges and opportunities”. In: *2015 International Conference on Big Data and Smart Computing (BIGCOMP)*. IEEE. 2015, pp. 271–278. DOI: 10.1109/35021BIGCOMP.2015.7072812.

[95] Emmanuel Pietriga. *IsaViz: a Visual Environment for Browsing and Authoring RDF Models*. 2002. URL: http://www.w3.org/2001/11/IsaViz/.

[96] Emmanuel Pietriga et al. “Fresnel: A Browser-independent Presentation Vocabulary for RDF”. In: *The Semantic Web - ISWC 2006 - 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5-9, 2006. Proceedings*. Athens, GA: Springer-Verlag, 2006, pp. 158–171. DOI: 10.1007/11926078_12.

[97] Laura Po et al. “Linked Data Visualization: Techniques, Tools, and Big Data”. In: *Synthesis Lectures on Semantic Web: Theory and Technology* 10.1 (2020), pp. 1–157. DOI: 10.2200/S00967ED1V01Y201911WBE019.

[98] Igor O. Popov et al. “Connecting the Dots: A Multi-pivot Approach to Data Exploration”. In: *The Semantic Web – ISWC 2011 – 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I*. Ed. by Lora Aroyo et al. ISWC ’11. Springer, 2011, pp. 553–568. DOI: 10.1007/978-3-642-25073-6.

[99] D. A. Quan and R. Karger. “How to Make a Semantic Web Browser”. In: *Proceedings of the 13th International Conference on World Wide Web*. WWW ’04. New York, NY, USA: Association for Computing Machinery, 2004, 255–265. ISBN: 158113844X. DOI: 10.1145/988672.988707. URL: https://doi.org/10.1145/988672.988707.

[100] Filip Radulovic et al. “A comprehensive quality model for linked data”. In: *Semantic Web* 9.1 (2018), pp. 3–24. DOI: 10.3233/SW-170267.

[101] Patrick Riehmann, Manfred Hanfler, and Bernd Froehlich. “Interactive sankey diagrams”. In: *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.* IEEE. 2005, pp. 233–240. DOI: 10.1109/INFVIS.2005.1532152.

[102] Laurens Rietveld and Rinke Hoekstra. “The YASGUI family of SPARQL clients 1”. In: *Semantic Web* 8.3 (2017), pp. 373–383. DOI: 10.3233/SW-150197.

[103] Lloyd Rutledge, Jacco van Ossenbruggen, and Lynda Hardman. “Making RDF presentable: Integrated global and local semantic web browsing”. In: *Proceedings of the 14th international conference on world wide web*. WWW ’05. Association for Computing Machinery, 2005, 199–206. DOI: 10.1145/1060745.1060777.

[104] Vedran Sabol et al. “Discovery and Visual Analysis of Linked Data for Humans”. In: *The Semantic Web – ISWC 2014*. Ed. by Peter Mika et al. Cham: Springer International Publishing, 2014, pp. 309–324. ISBN: 978-3-319-11964-9. DOI: 10.1007/978-3-319-11964-9_20.

[105] Henrique Santos et al. “From Data to City Indicators: A Knowledge Graph for Supporting Automatic Generation of Dashboards”. In: *The Semantic Web*. Ed. by Eva Blomqvist et al. Cham: Springer International Publishing, 2017, pp. 94–108. ISBN: 978-3-319-58451-5. DOI: 10.1007/978-3-319-58451-5_7.

[106] Arvind Satyanarayan et al. “Reactive Vega: A Streaming Dataflow Architecture for Declarative Interactive Visualization”. In: *IEEE Transactions on Visualization and Computer Graphics* 22.1 (2016), pp. 659–668. DOI: 10.1109/TVCG.2015.2467091.

[107] Mario Schmidt. “The Sankey diagram in energy and material flow management: Part I: History”. In: *Journal of industrial ecology* 12.1 (2008), pp. 82–94. DOI: 10.1111/j.1530-9290.2008.00004.x.

[108] m. c. schraefel et al. “The Evolving MSpace Platform: Leveraging the Semantic Web on the Trail of the Memex”. In: *Proceedings of the Sixteenth ACM Conference on Hypertext and Hypermedia*. HYPERTEXT ’05. Salzburg, Austria: Association for Computing Machinery, 2005, 174–183. DOI: 10.1145/1083356.1083391.

[109] m.c. schraefel and D. R. Karger. “The Pathetic Fallacy of RDF”. In: *SWUI 2006, the 3rd International Semantic Web User Interaction Workshop (colocated with ISWC2006), 6 Nov 2006, Athens, GA, USA*. 2006. URL: https://eprints.soton.ac.uk/262911/.

[110] *Scipy*. Accessed on April 25, 2020. URL: https://www.scipy.org/.

[111] Nigel Shadbolt and Kieron O’Hara. “Linked data in government”. In: *IEEE Internet Computing* 17.4 (2013), pp. 72–77. DOI: 10.1109/MIC.2013.72.

[112] Zeqian Sheny and Kwan-Liu Maz. “Path visualization for adjacency matrices”. In: *Proceedings of the 9th Joint Eurographics/IEEE VGTC conference on Visualization*. 2007, pp. 83–90.

[113] Ben Shneiderman. “The eyes have it: A task by data type taxonomy for information visualizations”. In: *Proceedings 1996 IEEE symposium on visual languages*. IEEE. 1996, pp. 336–343. DOI: 10.1109/VL.1996.545307.

[114] Ben Shneiderman. “The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations”. In: *VL ’96 – Proceedings of the IEEE Symposium on Visual Languages*. IEEE Computer Society, 1996, pp. 336–343. DOI: 10.1109/VL.1996.545307.

[115] Ben Shneiderman and Catherine Plaisant. “Strategies for Evaluating Information Visualization Tools: Multi-Dimensional in-Depth Long-Term Case Studies”. In: *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization*. BELIV ’06. Venice, Italy: Association for Computing Machinery, 2006, 1–7. ISBN: 1595935622. DOI: 10.1145/1168149.1168158.

[116] Claus Stadler, Michael Martin, and Sören Auer. “Exploring the Web of Spatial Data with Facete”. In: *Proceedings of the 23rd International Conference on World Wide Web*. WWW ’14 Companion. Seoul, Korea: Association for Computing Machinery, 2014, 175–178. ISBN: 9781450327459. DOI: 10.1145/2567948.2577022.

[117] Damian Steer. *BrownSauce: An RDF Browser*. 2003. URL: https://www.xml.com/pub/a/2003/02/05/brownsauce.html.

[118] *The story of inteGraality, or my quest to make it at the Wikimedia Hackathon*. Accessed on April 25, 2020. URL: https://commonists.wordpress.com/2019/07/06/the-story-of-integraality-or-my-quest-to-make-it-at-the-wikimedia-hackathon/.

[119] K. Thellmann et al. “LinkDaViz – Automatic Binding of Linked Data to Visualizations”. In: *The Semantic Web – ISWC 2015 – 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11–15, 2015, Proceedings, Part I*. Ed. by M. Arenas et al. Springer International Publishing, 2015, pp. 147–162. DOI: 10.1007/978-3-319-25007-6.

[120] Georgia Troullinou et al. “Exploring RDFS KBs using summaries”. In: *The semantic web – ISWC 2018*. Ed. by Denny Vrandečić et al. Springer International Publishing, 2018, 268–284. ISBN: 978-3-030-00671-6. DOI: 10.1007/978-3-030-00671-6_16.

[121] Georgia Troullinou et al. “Ontology understanding without tears: The summarization approach”. In: *Semantic Web* 8.6 (2017), pp. 797–815. DOI: 10.3233/SW-170264.

[122] Y. Tzitzikas, N. Manolis, and P. Papadakos. “Faceted Exploration of RDF/S Datasets: A Survey”. In: *Journal of Intelligent Information Systems* 48.2 (Apr. 2017), pp. 329–364. ISSN: 0925-9902. DOI: 10.1007/s10844-016-0413-8.

[123] *UMAP Learn*. Accessed on April 25, 2020. URL: https://umap-learn.readthedocs.io/en/latest/.

[124] *URIBurner*. 2009. URL: http://uriburner.com.

[125] Ahmet Uyar and Farouk Musa Aliyu. “Evaluating search features of google knowledge graph and bing satori”. In: *Online Information Review* (2015). DOI: 10.1108/oir-10-2014-0257.

[126] Corinna Vehlow, Fabian Beck, and Daniel Weiskopf. “The State of the Art in Visualizing Group Structures in Graphs.” In: *EuroVis (STARs)*. 2015, pp. 21–40. DOI: 10.2312/eurovisstar.20151110.

[127] Markel Vigo, Caroline Jay, and Robert Stevens. “Constructing Conceptual Knowledge Artefacts: Activity Patterns in the Ontology Authoring Process”. In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI ’15. Seoul, Republic of Korea: ACM, 2015, pp. 3385–3394. ISBN: 978-1-4503-3145-6. DOI: 10.1145/2702123.2702495.

[128] Markel Vigo, Caroline Jay, and Robert Stevens. “Design Insights for the next Wave Ontology Authoring Tools”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’14. New York, NY, USA: Association for Computing Machinery, 2014, 1555–1558. DOI: 10.1145/2556288.2557284.

[129] Hongwei Wang et al. “Ripplenet: Propagating user preferences on the knowledge graph for recommender systems”. In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 2018, pp. 417–426. DOI: 10.1145/3269206.3271739.

[130] Colin Ware et al. “Cognitive measurements of graph aesthetics”. In: *Information visualization* 1.2 (2002), pp. 103–110. DOI: 10.1057/palgrave.ivs.9500013.

[131] Paul Warren and Paul Mulholland. “Using SPARQL–the practitioners’ viewpoint”. In: *European Knowledge Acquisition Workshop*. Springer. 2018, pp. 485–500. DOI: 10.1007/978-3-030-03667-6_31.

[132] Marc Weise, Steffen Lohmann, and Florian Haag. “Ld-vowl: Extracting and visualizing schema information for linked data”. In: *2nd International Workshop on Visualization and Interaction for Ontologies and Linked Data*. 2016, pp. 120–127. URL: http://ceur-ws.org/Vol-1704/.

[133] R. White and R. Roth. *Exploratory Search: Beyond the Query-Response Paradigm*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool, 2013, p. 98. ISBN: 9781598297843. URL: https://ieeexplore.ieee.org/document/6812556.

[134] Avicenna Wisesa et al. “Wikidata Completeness Profiling Using ProWD”. In: *Proceedings of the 10th International Conference on Knowledge Capture*. K-CAP ’19. Marina Del Rey, CA, USA: Association for Computing Machinery, 2019, 123–130. ISBN: 9781450370080. DOI: 10.1145/3360901.3364425. URL: https://doi.org/10.1145/3360901.3364425.

[135] Kanit Wongsuphasawat et al. “Voyager: Exploratory Analysis via Faceted Browsing of Visualization Recommendations”. In: *IEEE Transactions on Visualization and Computer Graphics* 22.1 (2016), pp. 649–658. ISSN: 1077-2626. DOI: 10.1109/TVCG.2015.2467191.

[136] William A Woods. “What’s in a link: Foundations for semantic networks”. In: *Representation and understanding*. Elsevier, 1975, pp. 35–82. DOI: 10.1016/B978-0-12-108550-6.50007-0.

[137] Ka-Ping Yee et al. “Faceted metadata for image search and browsing”. In: *Proceedings of the SIGCHI conference on human factors in computing systems*. CHI ’03. Association for Computing Machinery, 2003, 401–408. DOI: 10.1145/642611.642681.

[138] Amrapali Zaveri et al. “Quality assessment for linked data: A survey”. In: *Semantic Web* 7.1 (2016), pp. 63–93. DOI: 10.3233/SW-150175.

[139] Amrapali Zaveri et al. “Using Linked Data to Evaluate the Impact of Research and Development in Europe: A Structural Equation Model”. In: *The Semantic Web – ISWC 2013*. Ed. by Harith Alani et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 244–259. ISBN: 978-3-642-41338-4. DOI: 10.1007/978-3-642-41338-4_16.

[140] Xiang Zhang, Gong Cheng, and Yuzhong Qu. “Ontology Summarization Based on Rdf Sentence Graph”. In: *Proceedings of the 16th International Conference on World Wide Web*. WWW ’07. New York, NY, USA: Association for Computing Machinery, 2007, 707–716. ISBN: 9781595936547. DOI: 10.1145/1242572.1242668. URL: https://doi.org/10.1145/1242572.1242668.

A

PRELIMINARY SKETCHES FOR S-PATHS

The very first sketches were specific to entities representing documents. They explored the idea of enriched timelines, maps or charts displaying additional attributes aggregate in sidebars (Fig. 61). In a second series of sketches, I generalised the principles to any kind of entities, adding more visualisations(Fig. 62,Fig. 63 and Fig. 64).

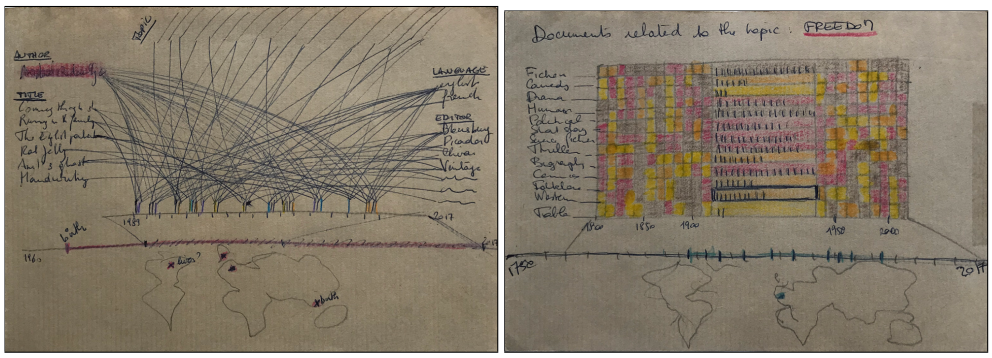


Figure 61: First series of sketches: left) entities typed as Documents on an enriched timeline, coordinated with a map; right) .

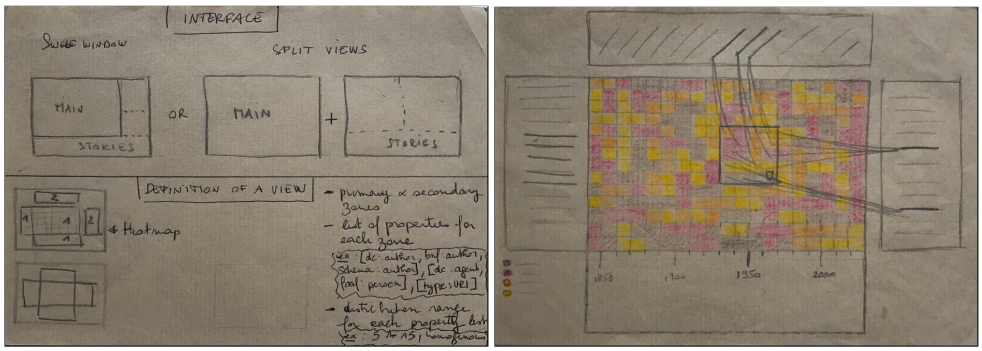


Figure 62: Early sketch, interface. Heatmap showing groups of entities organised by date on the x-axis, and a category on the y-axis, enriched by two other categories that will be highlighted when a selection is made (top and right bars)

The current state of the system differs from the sketches in several aspects. The templates are simpler, and do not make a systematic use of the sidebars. Although our framework theoretically supports the original templates, allowing to display an unlimited number of attributes, I implemented simple templates due to the weight of the queries. Our prototype does not allow zooming within a view to see more details about a group of entities, as we had first imag-

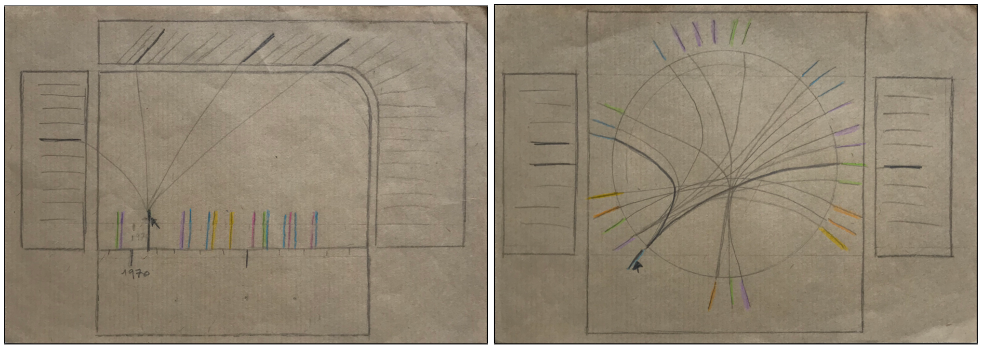


Figure 63: Timeline showing entities organised by date on the x-axis, enriched by two other categories that will be highlighted when a selection is made (left and top-right bars) Chord diagram showing entities and relations between them, enriched by two other categories that will be highlighted when a selection is made (left and right bars)

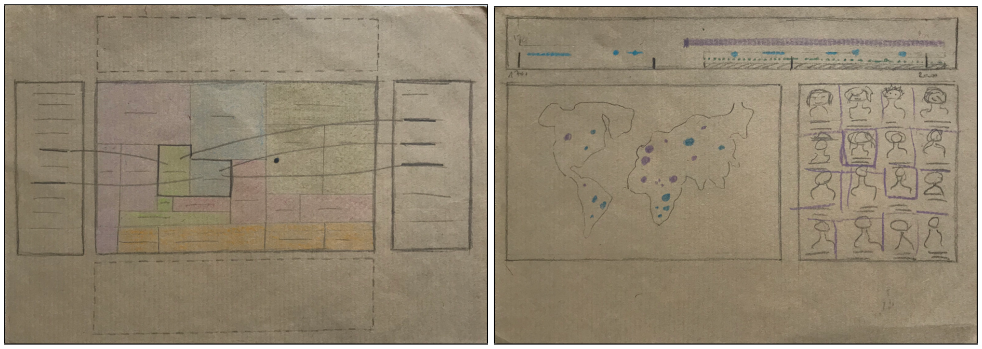


Figure 64: Hierarchical treemap showing groups and subgroups of entities, enriched by two other categories that will be highlighted when a selection is made (left and right bars). The top and bottom bars are not used because the system found no attributes to fill them. ”Story components”, complementary to the main component.

ined Fig. 61. A first version of the prototype did, but we realised that offering two different kinds of zoom — see details within the view, or inspect a subset in detail in another view — made the navigation too difficult to understand.

B

REPORTS OF THE WORKSHOPS CONDUCTED WITH RDF EXPERTS AND LAY USERS (IN FRENCH)

Those 2 workshops were organised by ILDA team for the French national Library. There were held in french, and the reports are in french.

- Workshop 1 . 26/02/2018 @inria Paris . 9 experts
- Workshop 2 . 27/032018 @BnF Paris . 7 lay users

scénarios courts

générique	exemple
trouver une partition pour des instruments et un nombre d'instrumentistes définis	chant tchèque accompagné au piano
se documenter sur un événement historique	les arguments pour et contre de l'affaire Dreyfus
trouver le décret de naturalisation d'une personne	
explorer un sujet	les femmes dans la littérature au XVIIIe s.
retracer l'histoire d'un titre de presse	
chercher une image pour illustrer un article, compatible avec la charte, avec des droits abordables, pas trop utilisée	article sur le thème du big data
chercher une image pour explorer une idée	les sourires dans les peintures du XVIIIe s.

scénarios longs

Groupe 1 : produire et partager de nouvelles connaissances

L'outil proposé permet de rechercher dans les données et de produire de nouvelles connaissances à partir de celles qui sont découvertes, et partager le résultat. Idée de coopération entre les algorithmes d'indexation et les utilisateurs.

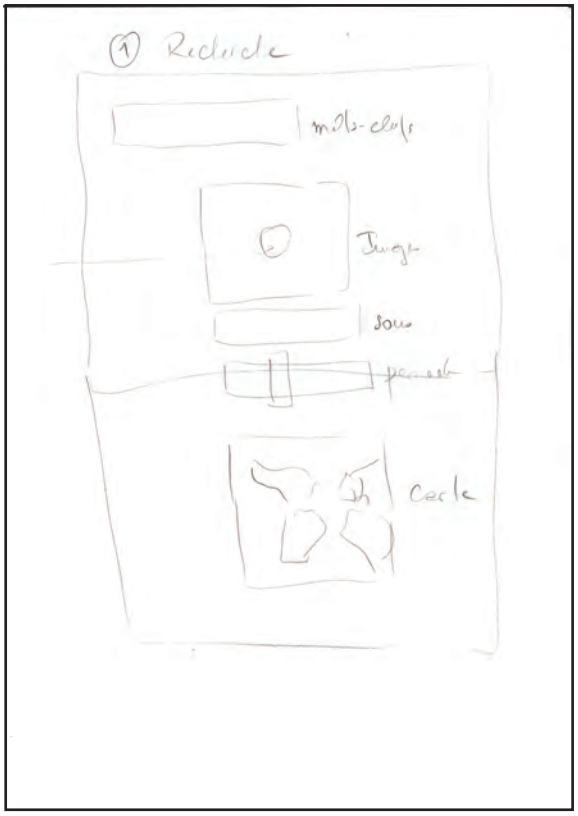
La recherche est multiple, par mot-clé, selon différents critères (temporel et géographique), ou par dépôt de fichier (détection des thématiques d'une image ou d'un son).

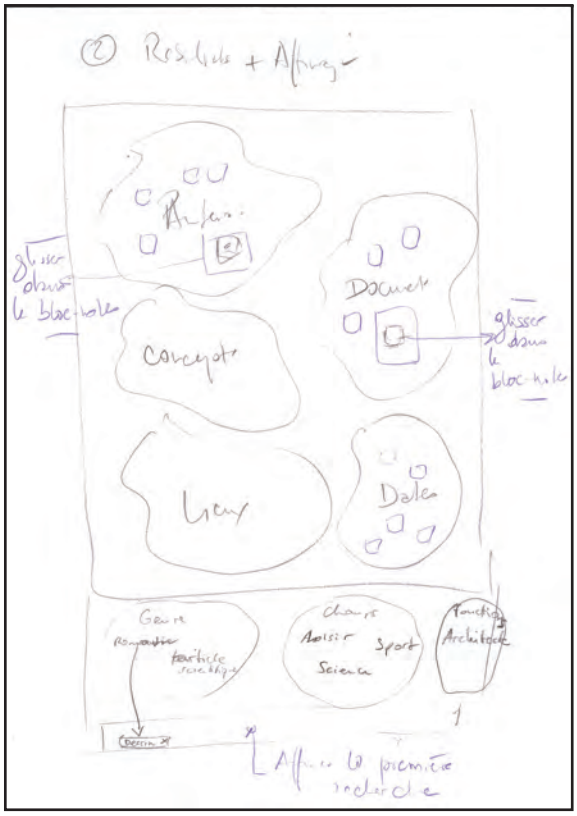
Les résultats sont regroupés par entités sémantiques (documents, auteurs, concepts, lieux, dates). Des facettes, sémantiques également (par exemple fonction des personnes), permettent d'affiner les résultats : on fait glisser les filtres dans une zone pour restreindre les résultats.

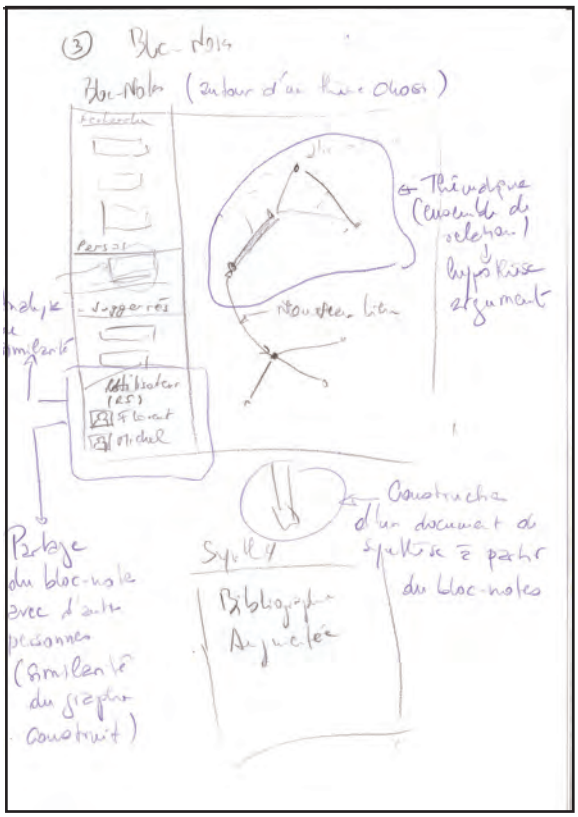
Lorsqu'on sélectionne un ensemble de documents, ils sont affichés à gauche, avec en vis à vis le graphe des relations qui existent entre elles. L'utilisateur peut ajouter des relations dans le graphe, ou créer une sélection et l'annoter. Les annotations faites par un utilisateur sont identifiables.

En bas de la colonne de gauche se trouve une zone de recommandations de contenus liés qui évolue avec la sélection de l'utilisateur. Parmi les recommandations, on trouve des contenus créés par d'autres utilisateurs. On peut les afficher en superposition pour voir ce qui est commun ou différent, et comparer les annotations.

Il est également possible de faire un export, sorte de bibliographie augmentée contenant les documents, les relations, et les annotations sur le graphe.


① Recherche
mots-clés
Image
Sons
Carte


② Résultats + Affinage
Auteurs
Concepts
Documents
Lieux
Dates
glisser dans le bloc-notes
Genre
Romantique
Classes
Loisir
Sport
Science
Affiner la première recherche


③ Bloc-notes
Bloc-Notes (autour d'un thème choisi)
Personnes
Documents
Bibliographie
Annotée
Partage du bloc-notes avec d'autres personnes (similarité du graphe construit)

Groupe 2 : recherche d'une image pour illustrer le thème du big data

On saisit une recherche plein texte. L'interface propose soit de voir les images, soit de voir d'autres publications sur le même thème (c'est une pratique courante de regarder ce qui se fait sur un sujet avant de choisir une image).

On va d'abord s'intéresser aux images. La présentation importe peu pour des experts, une liste suffit. Des facettes permettent d'affiner les résultats. Une première facette propose le champ sémantique identifié, avec les mots par ordre d'importance (du nombre de résultats). On peut également filtrer par couleur, par texture, par type de droits, et par nombre de téléchargements (nécessité de se démarquer, et donc d'éviter ce qui a déjà été utilisé). On a aussi besoin de filtrer par date du contenu de l'image. En effet, un coffre-fort pourrait être pertinent pour représenter le big data, mais pas un coffre-fort espagnol du au XVIIe s.

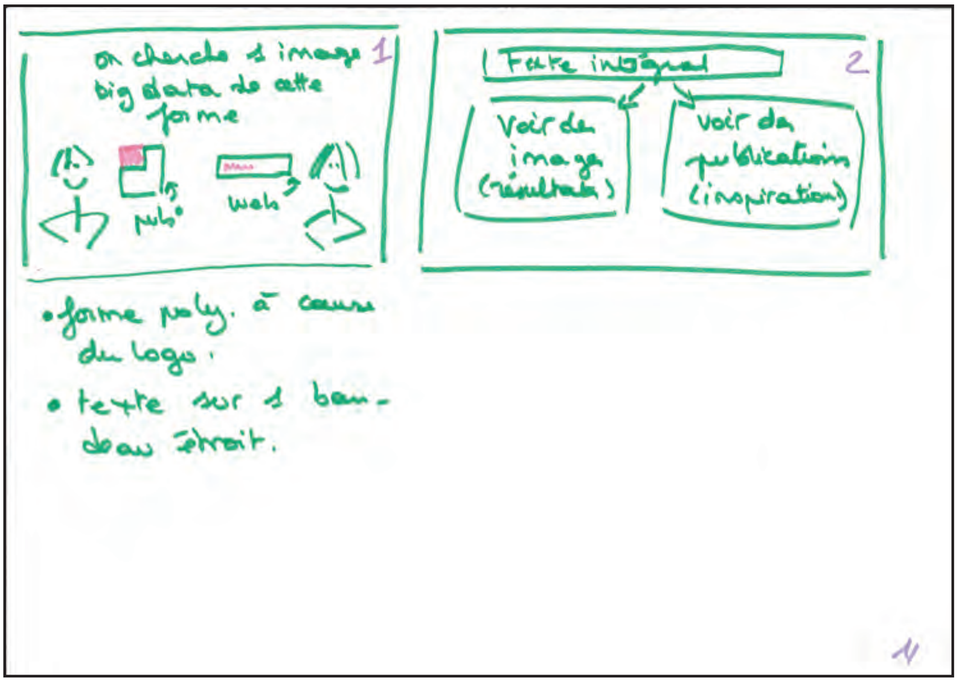
On souhaite pouvoir chercher une image en dessinant une forme qui correspond au cadrage imposé par la charte (superposition du logo, etc). Le système est capable de détecter si les éléments importants de l'image (personnes) ne sont pas dans le masque, et de ne pas afficher les résultats correspondants. En cas de doute, il propose l'image.

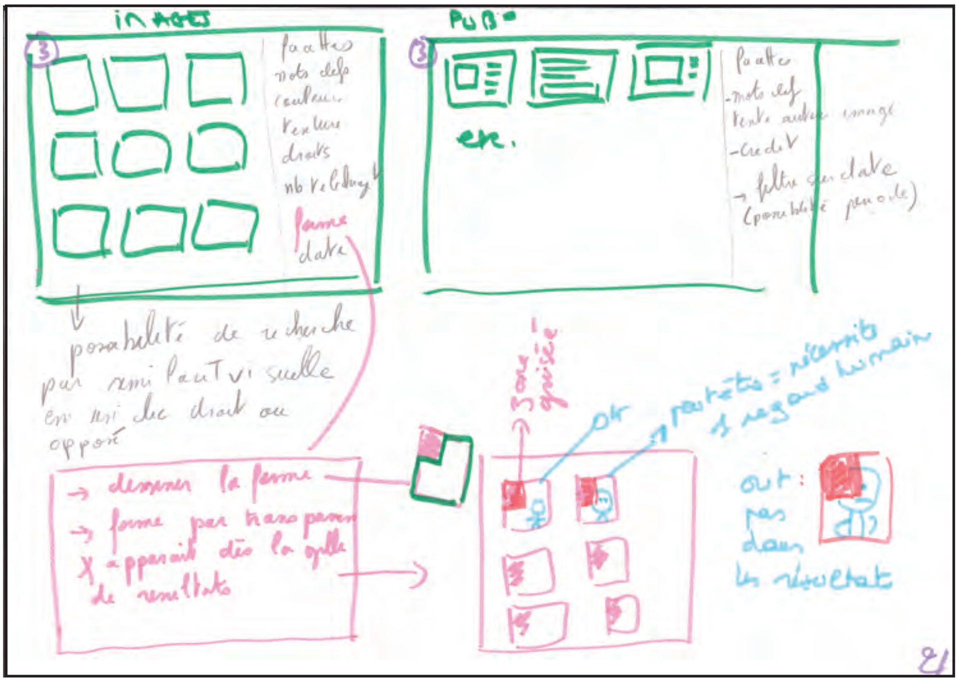
Il est possible de rebondir sur un résultat pour lancer une recherche sur tous les résultats similaires ou tous les résultats opposés, que ce soit depuis une notice image ou la liste de résultats.

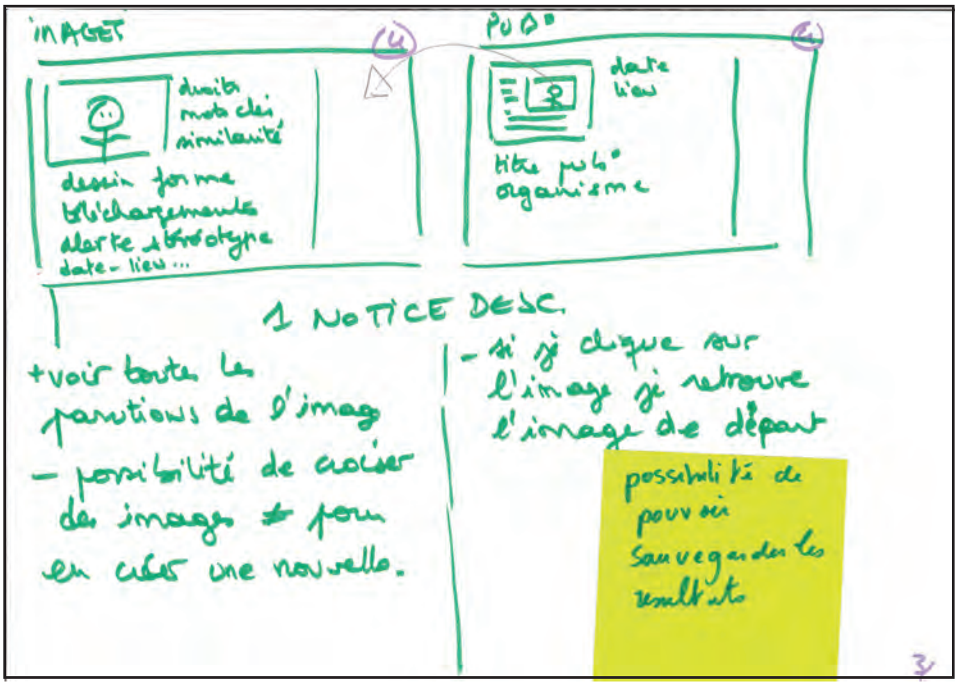
On a la possibilité de sauvegarder un set de résultats pour y revenir, ou de l'exporter.

L'affichage de la liste de publications (images en contexte) propose également des facettes avec le champ lexical et la source.

On peut imaginer une alerte stéréotype (par exemple pour la parité, lorsqu'il n'y a que des hommes sur une photo).


on cherche 1 image big data de cette forme
1
pub°
web
Texte intégral
2
Voir des images (résultats)
voir des publications (inspiration)
• forme poly. à cause du logo.
• texte sur 1 bandeau étroit.
1


IMAGES
3
facettes
mots clefs
couleurs
texture
droits
nb téléchargt
forme
date
PUB°
3
facettes
- mots clef
texte autour image
- crédit
→ filtre par date (possibilité période)
etc.
possibilité de recherche par similarité visuelle en lien des droits ou opposé
→ dessiner la forme
→ forme par transparence
X apparaît dès la grille de résultats
→ zone grisée
OK
portraits = nécessité 1 regard humain
out : pas dans les résultats
2


IMAGES
4
droits
mots clés
similarité
dessin forme
téléchargements
alerte stéréotype
date - lieu ...
PUB°
4
date
lieu
titre pub°
organisme
1 NOTICE DESC.
+ voir toutes les parutions de l'image
- possibilité de croiser des images pour en créer une nouvelle.
- si je clique sur l'image je retrouve l'image de départ
possibilité de pouvoir sauvegarder les résultats
3

Groupe 3 : rechercher des partitions en fonction de la distribution

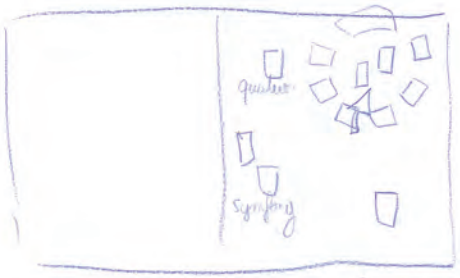
Deux musiciens souhaitent trouver une partition qu'ils peuvent jouer ensemble : un chant accompagné au piano.

Ils ont l'idée qu'un chant tchèque pourrait convenir, et cherchent les mots : chant tchèque piano. Les résultats proposés apparaissent sous forme d'une liste classique, avec en vis à vis avec un graphe destiné à préciser la recherche. Par défaut le graphe propose des clusters par type de résultat : partitions, livres et enregistrements. Si l'utilisateur sélectionne le groupe Partitions, le critère est pris en compte et le graphe est re-clusterisé selon un autre critère : ici l'effectif.

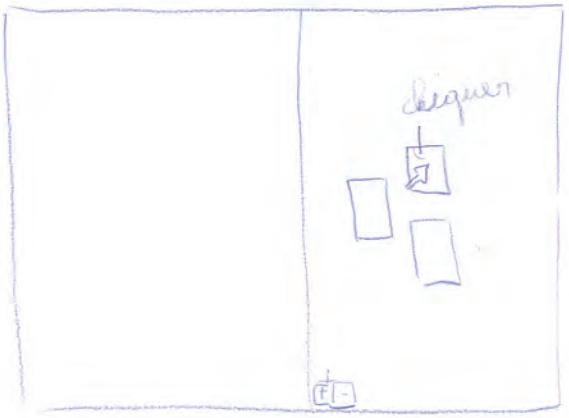
Au survol d'un élément (cluster ou sous-élément du cluster), les propriétés associées apparaissent dans une "roue" qui permet de rebondir sur ces critères de façon fine : ajouter / exclure des critères de recherche, consulter un ensemble lié.

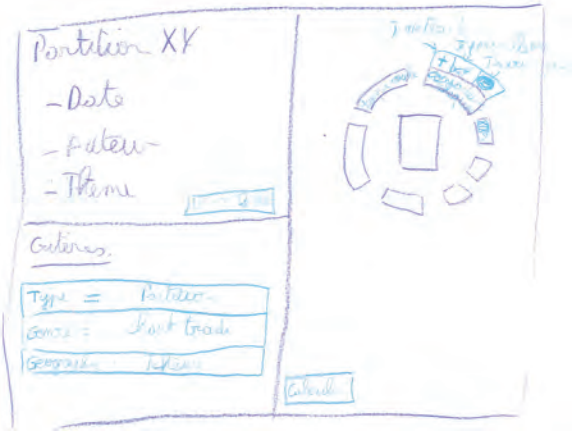
Idée de laisser visible en transparence un graphe plus grand que celui pertinent pour la recherche, une sorte de contexte, pour permettre à l'utilisateur de continuer la navigation, d'élargir ou réorienter sa recherche.

- On zoome dans le cluster partition
- Ce recluster les résultats

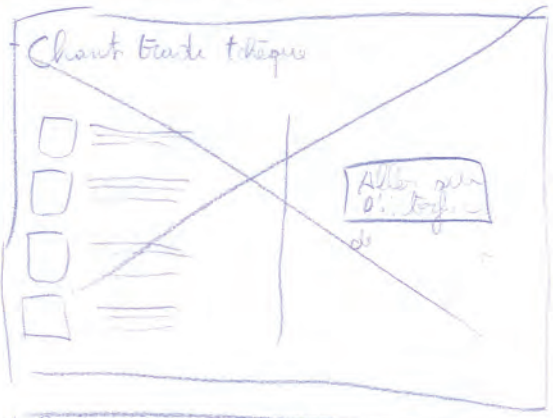


- La roue apparait au survol d'un groupe
- On rezoome sur un dernier

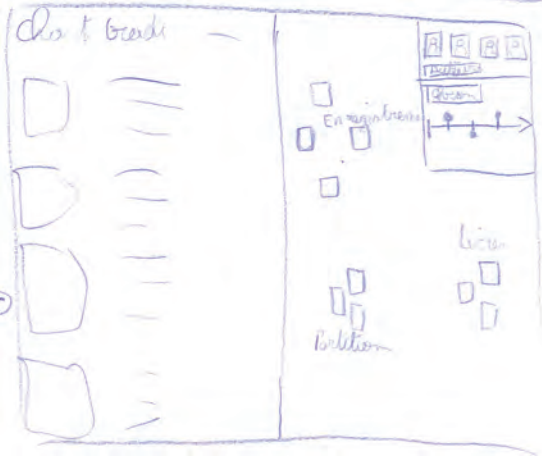




- Tiens regarde, il y a l'air d'y avoir plein de résultats de chants Grégorien ad. Cliquons
- Visiblement, on a trois groupes dans ce chant
- Regardons les partitions



N'importe quel cluster génère une roue →



Groupe 4 : Ok data, médiation semi-automatique et langue naturelle

OK Data est un outil inspiré de Ok Google, qui traduit les recherches des utilisateurs et leurs résultats en langage naturel.

La partie gauche de l'interface amène l'utilisateur à préciser sa recherche par des questions destinées à désambigüiser les termes qu'il a utilisés (polysémie), identifier le but de la recherche (découverte, pédagogique, administratif, technique), et préciser le type de documents recherchés.

La seconde partie est dédiée à l'affichage des résultats. Ceux-ci sont éditorialisés de façon automatique : proposition de documents recommandés, de contenus de médiation, de cartes, de frises...

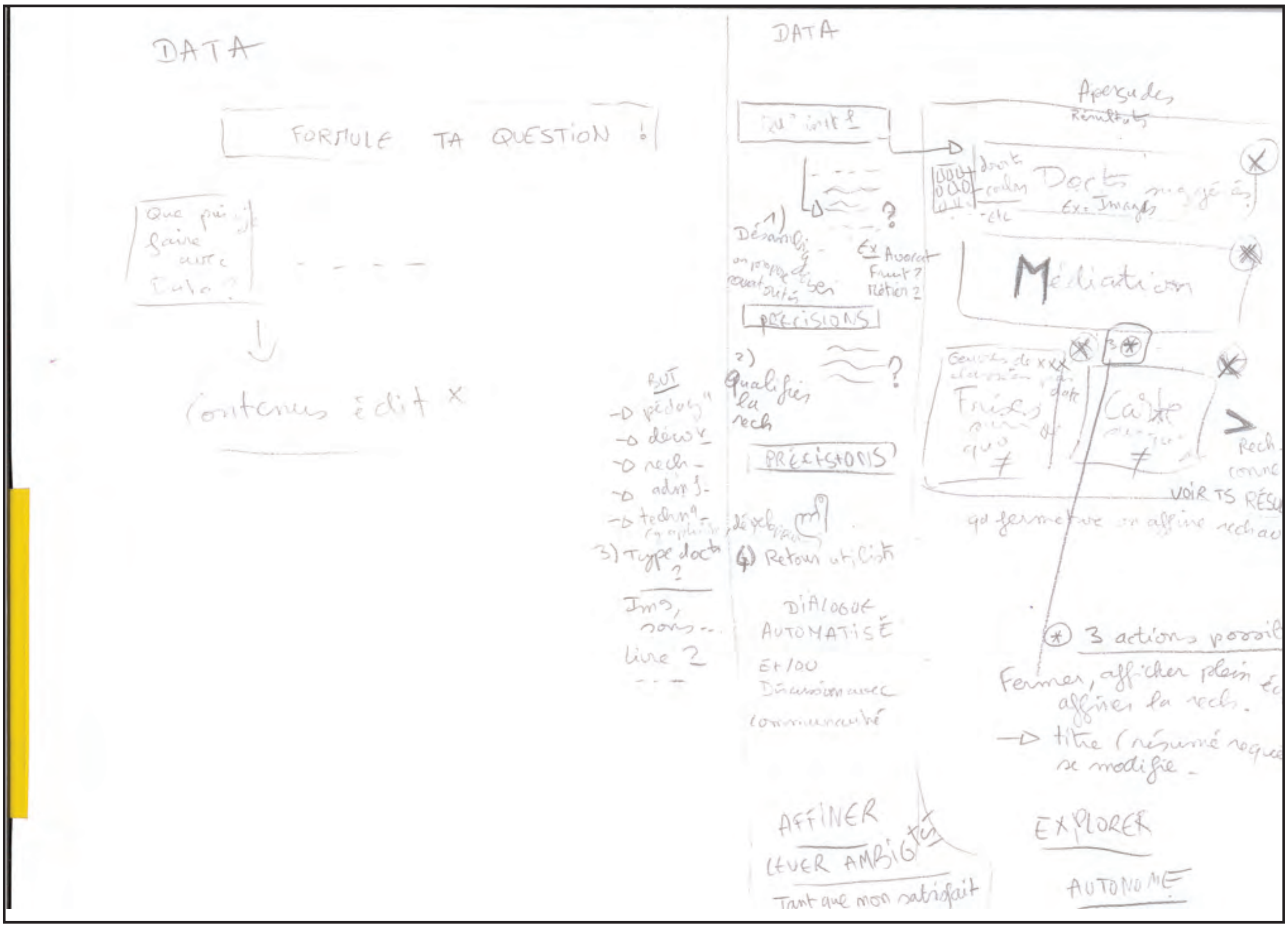
Par exemple une recherche avec les mots clé "oeuvres femmes XVIIIe s." pourra être interprétée comme suit :
- oeuvres écrites au XVIIIe par des femmes
- oeuvres qui parlent de la condition des femmes au XVIIIe
- publications éditées par des femmes au XVIIIe siècle
- portraits de femmes dessinés au XVIIIe siècle
- dossiers pédagogiques de la BnF qui parlent des femmes

Une visualisation sera proposée pour chaque interprétation, présentée selon les critères qui semblent pertinents en fonction des résultats (géographique, chronologique...).

Pour tout contenu 3 actions possibles:
- le supprimer,
- le passer en plein écran pour poursuivre l'exploration
- affiner les critères avec les filtres de tri proposés.

L'outil offre une assistance à l'utilisateur pour éviter qu'il ne soit pas perdu devant l'infini des possibles, tout en lui permettant de prendre la main et d'explorer librement,

L'utilisateur peut évaluer la pertinence d'un résultat, de façon à ce que le système puisse apprendre des scénarios d'échec.


DATA
FORMULE TA QUESTION !
Que puis-je faire avec Data ?
Contenus édit x
DATA
Qu'est-ce ?
1) Désambiguïser
Ex Avocat Fruit ? Métier ?
PRÉCISIONS
2) Qualifier la rech
BUT
-> pédag
-> décou
-> rech.
-> admin.
-> techn
3) Type docts ?
Ims, sons...
Livre ?
PRÉCISIONS
4) Retour utilisateur
DIALOGUE AUTOMATISÉ
Et/OU
Discussion avec communauté
AFFINER
LEVER AMBIGUÏTÉ
Tant que non satisfait
Aperçu des Résultats
Docts suggérés
Ex. Images
Médiation
Caisse
VOIR TS RÉSU
3 actions possibles
Fermer, afficher plein écran, affiner la rech.
-> titre / résumé requête se modifie.
EXPLORER
AUTONOME

scénarios courts

Caroline, chercheuse, souhaite faire un état de l'art sur les graphes multivariés afin de pouvoir situer son travail et écrire un papier original.

Nicole, chercheuse, veut se documenter sur les femmes d'affaire au XVIe et XVIIe en France.

Hugues, amateur de peinture, voudrait voir tous les autoportraits faits par des femmes peintres à travers l'histoire.

Mélissa, bibliothécaire, cherche des documents de toute sorte liés au compositeur Haydn, en vue de réaliser une cartographie du sujet.

Martine, passionnée de musique, cherche tous les romans qui évoquent des œuvres musicales, avec un pointeur vers les passages où elles sont évoquées.

Hugues, professionnel du web, cherche un espace de discussion autour des générateurs de sites statiques pour connaître les nouveautés, discuter, éventuellement demander de l'aide à la communauté et identifier les évènements communautaires liés au sujet.

Nicole, chercheuse, cherche tous les éditeurs et éditrices du XIXe siècle en France en vue de consolider une base de données dans le cadre d'un projet ANR.

Juliette, commissaire d'exposition, cherche des pochettes de disques pour une exposition sur le thème de l'immigration.

Hugo, développeur, cherche des gravures de presse open source, qui soient belles et dans un format exploitable, pour illustrer un projet sur lequel il travaille.

Valentin voudrait savoir quels artistes étaient présents au moment où Mozart a composé sa première symphonie (Londres 1764).

Philippe souhaite savoir combien de fois un certain papier a été cité.

Philippe, ingénieur pédagogique, veut construire des parcours de ressources pédagogiques categorisées (en termes de pédagogie).

scénarios longs

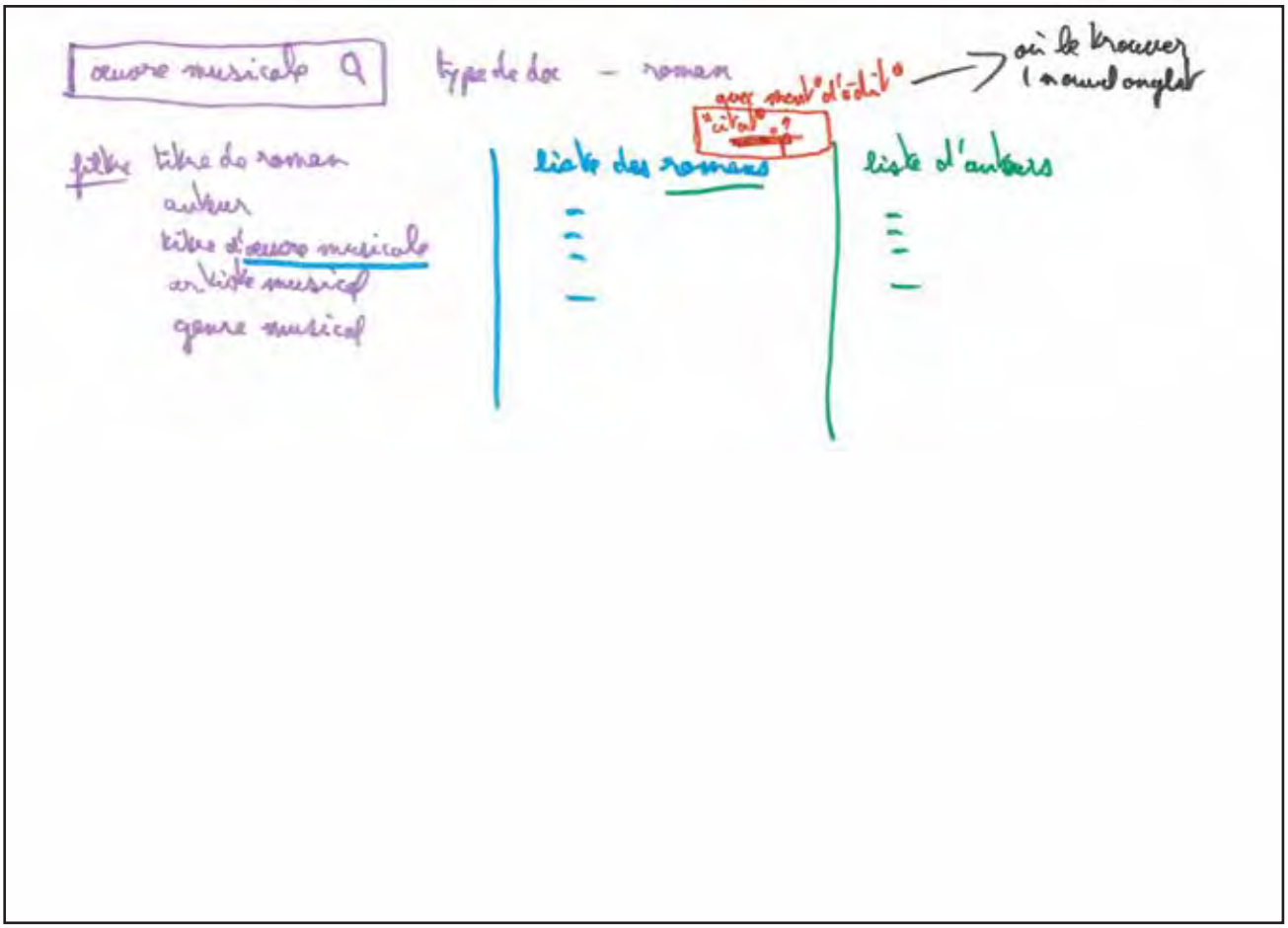
1 . Réponse au scénario : *Martine, passionnée de musique, cherche tous les romans qui évoquent des œuvres musicales, avec un pointeur vers les passages où elles sont évoquées.*

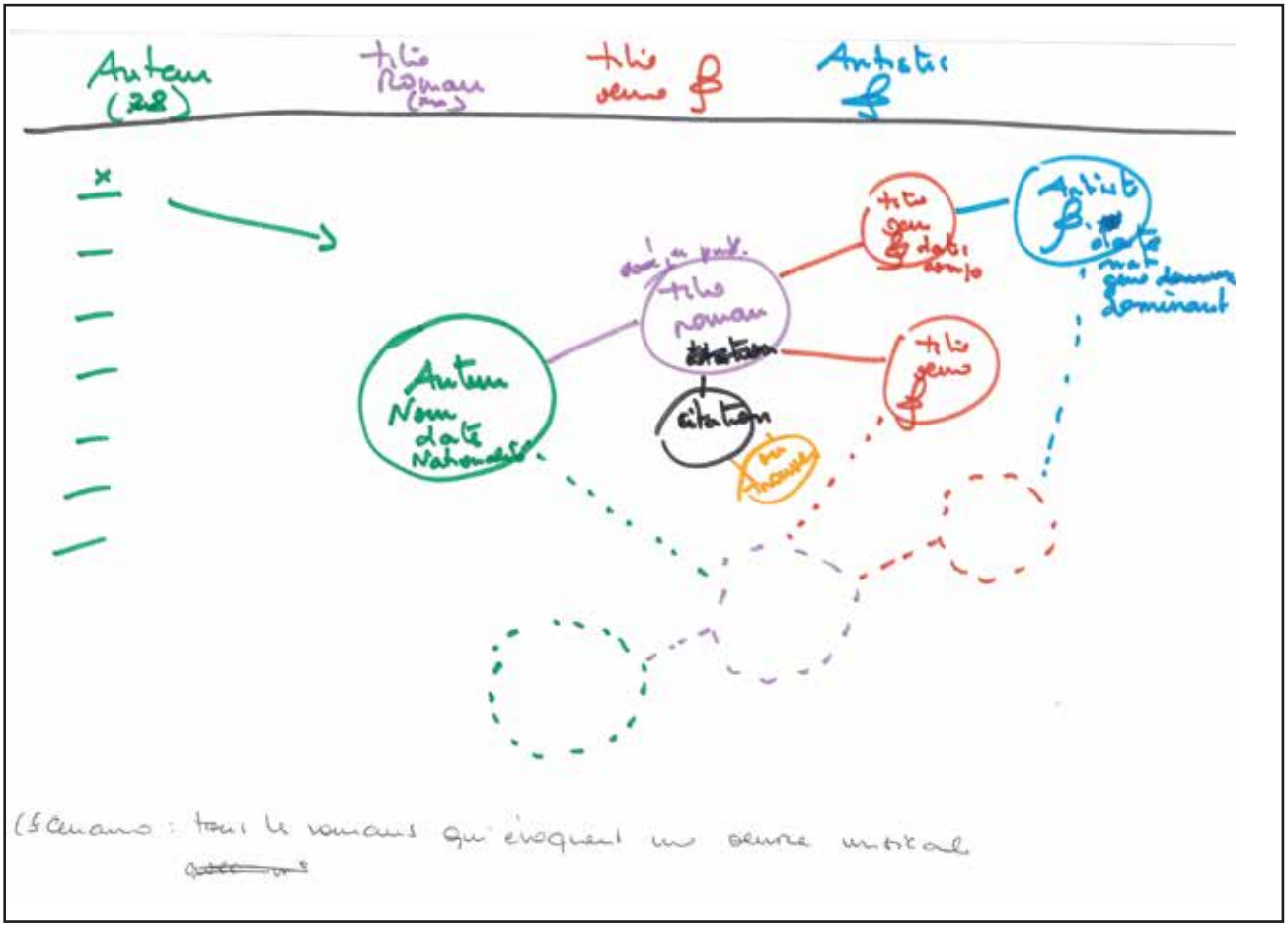
La première interface proposée a trois colonnes, chacune sous forme de liste :
La colonne de gauche liste tous les filtres :
- titre du roman
- auteur
- titre de l'oeuvre musicale
- artiste (compositeur / groupe / interprète)
- genre musical

Lorsque que l'on clique sur un filtre les valeurs correspondantes s'affichent dans la seconde ou la troisième colonne. Au survol sur une valeur le passage contenant la mention de l'oeuvre s'affiche dans une info bulle. Au clic on montre les mentions d'édition, et notamment le numéro de page.

La seconde interface est plus graphique. Dans un menu horizontal en haut de page on a les filtres auteur, titre du roman, titre de l'oeuvre et artiste.

Le clic sur l'un des filtres, ici les auteurs, affiche la liste de valeurs correspondantes à gauche de l'interface. Si l'on sélectionne une de ces valeurs un graphe est proposé en regard. Il part de la ressource sélectionnée, et montre quelques informations sur l'auteur comme la nationalité, les dates de naissance et de mort, ainsi que tous les romans, œuvres et artistes liés à l'auteur, et les liens de chacune de ces ressources entre elles.





(Scénario : tous les romans qui évoquent une œuvre musicale

2 . Réponse au scénario : *Hugues, professionnel du web, cherche un espace de discussion autour des générateurs de sites statiques pour connaître les nouveautés, discuter, éventuellement demander de l'aide à la communauté et identifier les évènements communautaires liés au sujet*.

Le point d'entrée de l'interface est un champ de recherche avec auto-completion.

Le groupe a identifié différents types de ressources : ressources documentaires, logiciels, logiciels dérivés, outils, exemples de projets, évènements : meetups, conférences, ateliers, canaux de discussion, personnes : experts, médiateurs...

L'interface présentée montre les résultats d'une recherche, en se concentrant sur les personnes / la communauté.
Les informations identifiées comme potentiellement utiles sont le nom d'une personne, son activité professionnelle, depuis quand elle travaille sur tel ou tel sujet, ou sur des sujets proches, son employabilité... En l'occurence on peut voir si les personnes sont connectées, leur indice de popularité ainsi que des tags (thèmes, compétences, savoirs-faire...)
Lorsqu'on sélectionne une personne, on a accès à plusieurs listes de ressources :
- les projets ou outils auxquels elle a participé (titre et jauge)
- les contenus qu'elle a recommandés ou aimés
- des informations la concernant (hobbies, etc)
- ses contacts

Le groupe évoque l'idée, non développée dans le prototype, d'une représentation sous forme de graphe qui permettrait de représenter graphiquement la taille de la communauté, son niveau d'activité, et de la comparer à d'autres communautés.

Ressources

- documentation (manuel / tuto)
- produit (logiciels, ~~logiciels~~ et dérivé)
- exemples de projets

Flux

- chat
- twitter
- Réseau sociaux ...

Événements

- Meetup/rencontres
- Conférences
- Atelier / Workshop
-

Personnes / communauté

- taille ~~des~~
- identité
- activité
- historique
- qualité
- employabilité
- disponibilité
- "Clés d'entrée" (Modérateurtrices)
- experts
- interaction

Générateur de sites statiques R

Personnage

- Hugues
  Dev | UX | lang
- Nicole
- Leo
- ...

B+ D A

générateur de site statique Q

La communauté

Machin gère le tchat + répond aux questions GNENTICS

+ a recommandé
- un truc
- la doc

V le tricent

Forum

Tchat

+ participe
- a un logiciel
- a la doc

+ et en plus
- il fait de la musique

et il connait tout le monde 53 contacts

a aimé...

travaille sur...

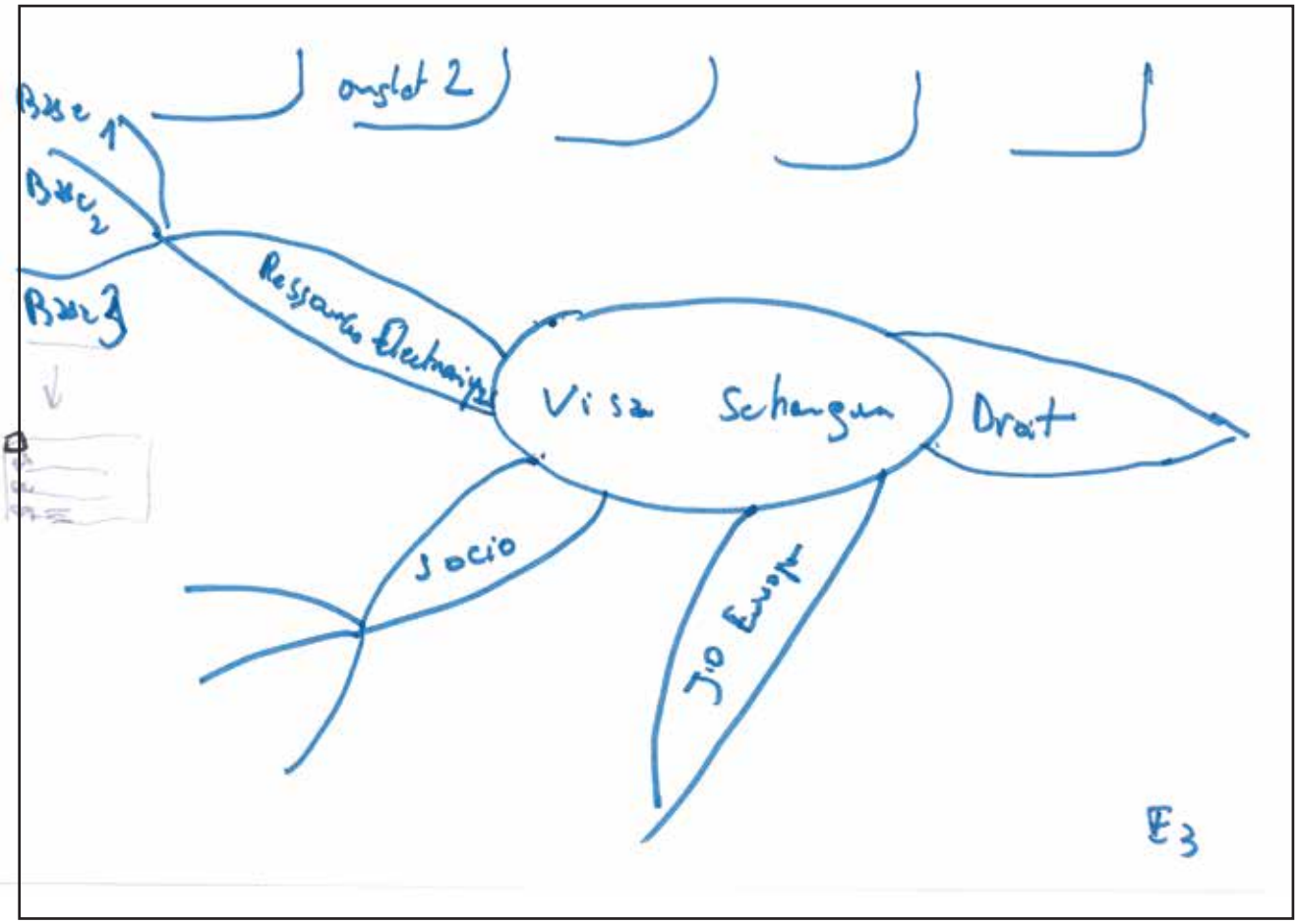
3 . Réponse au souhait de pouvoir mettre en commun des documents de provenance différentes (venant d'une bibliothèque, d'une communauté, ou de grandes bases de données comme Francis et Pascal...), et d'avoir accès au classement fait par les autres utilisateurs.

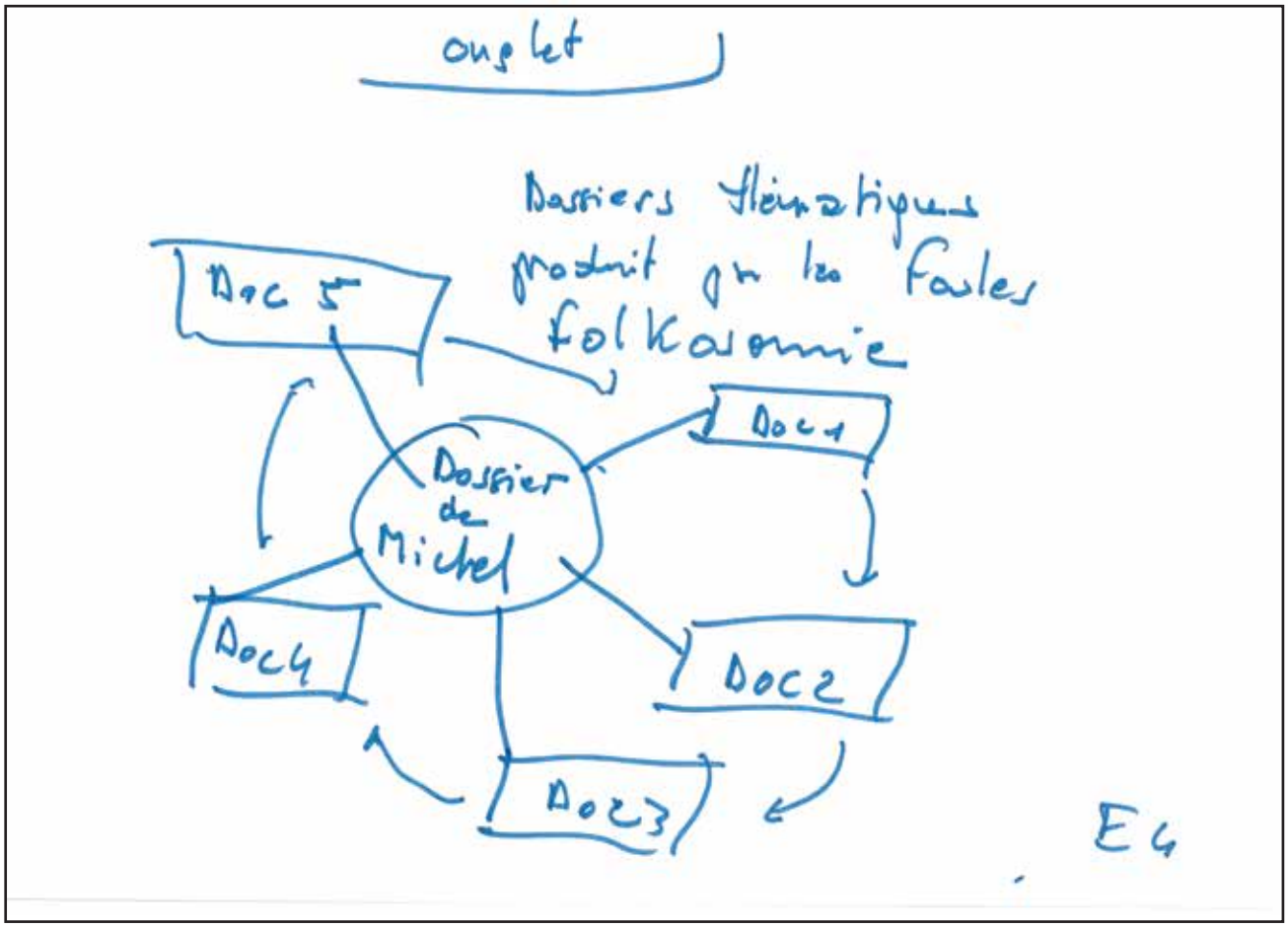
L'utilisateur fait une recherche. Les résultats sont présentés dans trois onglets : une liste classique, une carte mentale, et un accès aux dossiers d'autres utilisateurs.

La carte mentale, inspirée de wikimap, représente une hiérarchie par domaine. Par exemple sur une recherche "schengen", les domaine pourraient être : droit, sociologie, presse, etc. Le groupe souligne l'importance pour l'utilisateur de pouvoir identifier rapidement ce qui ne l'intéresse pas afin de l'ignorer : sorte de sélection inversée.

Les dossiers des autres utilisateurs sont consultables un par un. Les documents sont classés selon les tags attribués par l'utilisateur (principe de la folsksonomy), et présentés sous forme de graphe. On reprend le principe d'outils comme Pearltree ou Zotero.

Par défaut, le résultat de la recherche donne un aperçu de ce qui se trouve dans chacun des trois onglets.


onglet 2
Base 1
Base 2
Base 3
Ressources Electroniques
Visa Schengen
Droit
Socio
J.O Europ
E3


onglet
Dossiers thématiques
produit par les fiches
folksonomie
Doc 5
Doc 1
Dossier de Michel
Doc 4
Doc 2
Doc 3
E4

C

EVALUATION OF S-PATHS — FINAL QUESTIONNAIRE

1. How would you rate your overall knowledge about Nobel prizes after this experiment?
   - ☐ Very poor
   - ☐ Poor
   - ☐ Good
   - ☐ Excellent
2. Please answer the following questions:
   a) How many Nobel categories?
      - ☐ 2
      - ☐ 4
      - ☐ 6

   b) How many Nobel laureates?
      - ☐ 45
      - ☐ 911
      - ☐ 1378
      - ☐ 2458

   c) How many of them are organizations?
      - ☐ 3
      - ☐ 11
      - ☐ 23
      - ☐ 37

   d) What is the proportion of female laureates?
      - ☐ 1
      - ☐ 4
      - ☐ 12
      - ☐ 25

   e) Over which time range have Nobel prizes existed?
3. Which questions would you have been able to answer before the experiment?
4. Have you learnt any other fact?

D

EVALUATION OF PATH OUTLINES — TASKS

D.1 NOBEL DATASET

1. Consider all the awards in the dataset. For what percentage of them can you find the label of the birth place of the laureate of an award?

2. Consider all the laureates in the dataset. Imagine you want to plot a timeline: find all the paths of depth 1 or 2 starting from them and leading to a temporal information. Indicate the datatype of the values at the end of the path.

3. Imagine you want to plot a map of the universities. The most precise geographical information about the universities in the dataset seems to be the cities, which are aligned to Dbpedia through similarity links owl:sameAs. Find one or several properties in Dbpedia (http://dbpedia.org/sparql) that could help you place the cities on a map. Please tell us the reason that made you select these specific ones.

D.2 PERSEE DATASET

1. Consider all the documents in the dataset. For what percentage of them can you find the webpage of the photographer of an illustration of a document?

2. Consider all the articles in the dataset. Imagine you want to plot a timeline: find all the paths of depth 1 or 2 starting from them and leading to a temporal information. Indicate the datatype of the values at the end of the path.

3. Consider all the laureates in the dataset. Imagine you want to plot a timeline: find all the paths of depth 1 or 2 starting from them and leading to a temporal information. Indicate the datatype of the values at the end of the path.

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Exploration visuelle interactive de Graphes de Connaissance basée sur les chemins

Mots clés: Graphes de connaissance, Visualisation, RDF, Exploration Interactive, Web Sémantique, Linked Data

Résumé: Les Graphes de Connaissances représentent, connectent, et rendent interprétables par des algorithmes des connaissances issues de différents domaines. Ils reposent sur des énoncés simples que l'on peut chaîner pour former des énoncés de plus haut niveau. Produire des interfaces visuelles interactives pour explorer des collections dans ces données est un problème complexe, en grande partie non résolu. Dans cette thèse, je propose le concept de profils de chemins pour décrire les énoncés de haut niveau. Je l'utilise pour développer 3 outils open source: S-Paths permet de naviguer dans des collections a travers des vues synthétiques; Path Outlines permet aux producteurs de données de parcourir les énoncés qui peuvent produits par leurs graphes; et The Missing Path leur permet d'analyser l'incomplétude de leurs données. Je montre que le concept, en plus de supporter des interfaces visuelles interactives pour les graphes de connaissances, aide aussi a en améliorer la qualité.

Title: Path-Based Interactive Visual Exploration of Knowledge Graphs

Keywords: Knowledge Graphs, Visualisation, RDF, Interactive Exploration, Semantic Web, Linked Data

Abstract: Knowledge Graphs facilitate the pooling and sharing of information from different domains. They rely on small units of information named triples that can be combined to form higher-level statements. Producing interactive visual interfaces to explore collections in Knowledge Graphs is a complex problem, mostly unresolved. In this thesis, I introduce the concept of path outlines to encode aggregate information relative to a chain of triples. I demonstrate 3 applications of the concept with the design and implementation of 3 open source tools. S-Paths lets users browse meaningful overviews of collections; Path Outlines supports data producers in browsing the statements that can be produced from their data; and The Missing Path supports data producers in analysing incompleteness in their data. I show that the concept not only supports interactive visual interfaces for Knowledge Graphs but also helps better their quality.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France