CONCEPTS AND TOOLS FOR THE EFFECTIVE AND EFFICIENT USE OF WEB ARCHIVES

Von der Fakultät für Elektrotechnik und Informatik
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des akademischen Grades

DOKTOR DER NATURWISSENSCHAFTEN

Dr. rer. nat.

genehmigte Dissertation
von

Helge Holzmann, M.Sc.

geboren am 4. Mai 1986, in Celle, Deutschland

Hannover, Deutschland, 4. Februar 2019

Referent: Prof. Dr. techn. Wolfgang Nejdl
Korreferent: Prof. dr. ir. Arjen P. de Vries
Tag der Promotion: 04.02.2019

ABSTRACT

Web archives constitute valuable sources for researchers in various disciplines. However, their sheer size, the typically broad scope and their temporal dimension make them difficult to work with. We have identified three views to access and explore Web archives from different perspectives: user-, data- and graph-centric.

The natural way to look at the information in a Web archive is through a Web browser, just like the live Web is consumed. This is what we consider the *user-centric view*. The most commonly used tool to access a Web archive this way is the Wayback Machine, the Internet Archive's replay tool to render archived webpages. To facilitate the discovery of a page if the URL or timestamp of interest is unknown, we propose effective approaches to search Web archives by keyword with a temporal dimension through social bookmarks and labeled hyperlinks. Another way for users to find and access archived pages is past information on the current Web that is linked to the corresponding evidence in a Web archive. A presented tool for this purpose ensures coherent archived states of webpages related to a common object as rich temporal representations to be referenced and shared.

Besides accessing a Web archive by closely reading individual pages like users do, distant reading methods enable analyzing archival collections at scale. This *data-centric view* enables analysis of the Web and its dynamics itself as well as the contents of archived pages. We address both angles: 1. by presenting a retrospective analysis of crawl metadata on the size, age and growth of a Web dataset, 2. by proposing a programming framework for efficiently processing archival collections. ArchiveSpark operates on standard formats to build research corpora from Web archives and facilitates the process of filtering as well as data extraction and derivation at scale.

The third perspective is what we call the *graph-centric view*. Here, websites, pages or extracted facts are considered nodes in a graph. Links among pages or the extracted information are represented by edges in the graph. This structural perspective conveys an overview of the holdings and connections between contained resources and information. While this enables novel concepts of exploring Web archives, it also raises new challenges.

We present the latest achievements in all three views as well as synergies among them. For instance, important websites that can be identified from the graph-centric perspective may be of particular interest for the users of a Web archive. The data-centric view is used in both ways, it benefits from the graph-centric view to guide data studies but is also employed to prepare the data for the other views, like extracting graphs from archival collections. Finally, by considering the three views as different zoom levels of the same Web archive, they can be integrated in a holistic data analysis pipeline.

Keywords: *Web archives, temporal search, distributed data processing, Web analysis*

ZUSAMMENFASSUNG

Web-Archive stellen wertvolle Datenquellen für Forscher unterschiedlicher Disziplinen dar. Ihre schiere Größe, die typischerweise große Bandbreite an Daten sowie ihre zeitliche Dimension führen jedoch dazu, dass es nicht einfach ist, mit ihnen zu arbeiten. Um dies näher zu untersuchen, haben wir drei Sichtweisen auf den Zugriff und die Exploration von Web-Archiven identifiziert: Nutzer-, Daten- und Graphen-zentriert.

Ähnlich wie das Live-Web, ist der natürliche Weg die Informationen in einem Web-Archiv zu betrachten, durch einen Web-Browser. In dieser *Nutzer-zentrierten Sicht*, stellt die Wayback Machine des Internet Archives das bekannteste Tool zur Anzeige archivierter Webseiten dar. Um dabei das Auffinden solcher Seiten zu unterstützen, zu denen entweder die URL oder der gewünschte Zeitpunkt nicht bekannt sind, stellen wir einen effektiven Ansatz vor, um Web-Archive basierend auf sozialen Lesezeichen oder Hyperlinks zeitlich nach Schlüsselwörtern zu durchsuchen. Eine Alternative dazu sind zeitliche Informationen im Live-Web, die mit entsprechenden Belegen in einem Web-Archiv verlinkt sind. Unser dafür präsentierter Ansatz stellt sicher, dass zusammengehörige Seiten gemeinsam archiviert werden und somit als zeitliche Abbildung der durch sie repräsentierten Objekte dienen.

Neben der individuellen Betrachtung einzelner Webseiten durch den Nutzer, ermöglicht das sogenannte Distant-Reading Analysen im großen Stil. Die *Daten-zentrierte Sicht* betrachtet dabei sowohl das Web selbst, mit seinen dynamischen Eigenschaften, als auch die Inhalte der archivierten Seiten. Wir beschäftigen uns hierbei mit beiden Blickwinkeln: 1. indem wir Crawl-Metadaten in Bezug auf die Größe, das Alter sowie das Wachstum einer Webkollektion untersuchen, 2. indem wir ein Programmier-Framework zur effizienten Datenverarbeitung von Archiven vorstellen. ArchiveSpark arbeitet dabei ausschließlich mit Standardformaten, woraus Forschungskorpora durch die Anwendung von Filtern und die Extraktion von Daten erstellt werden können.

Aus der dritten Perspektive, der *Graphen-zentrierte Sicht*, werden Webseiten oder enthaltene Informationen als Knoten in einem Graphen betrachtet. Links oder Verbindungen zwischen diesen Daten werden durch Kanten in dem Graphen repräsentiert. Diese strukturelle Perspektive vermittelt dadurch einen Überblick über die im Archiv enthaltenen Elemente und deren Beziehungen. Diese Betrachtung der Daten ermöglicht neuartige Konzepte zur Erkundung von Web-Archiven. Gleichzeitig wirft sie jedoch auch neue Fragen auf.

Neben den neuesten Ergebnissen aus allen drei Sichtweisen präsentieren wir auch die Synergien zwischen diesen. Beispielsweise hilft die Graphen-zentrierte Sicht dabei, wichtige Webseiten zu identifizieren, die für Nutzer von besonderem Interesse sein können. Die Daten-zentrierte Sicht profitiert einerseits ebenfalls von der Graphen-zentrierte Sicht, um Webseiten mit hoher Relevanz für eine Datenanalyse ausfindig zu machen, andererseits wird sie genutzt, um die benötigten Daten für die beiden anderen Perspektiven aufzubereiten, zum Beispiel zur Konstruktion eines Graphen basierend auf den archivierten Daten. Weiterhin können die drei Sichtweisen als unterschiedliche Zoomstufen auf ein und dieselben Daten angesehen werden, was sich vor allem bei Datenanalyse-Prozessen als sehr sinnvoll erwiesen hat.

Schlagwörter: *Web-Archive, zeitliche Suche, verteilte Datenverarbeitung, Web-Analyse*

ACKNOWLEDGMENTS

Throughout the course of my PhD studies, I have worked with many great researchers, supervisors, colleagues, collaborators and partners, who have supported me in various respects. The biggest thanks goes to Prof. Dr. Wolfgang Nejdl for his guidance, help and the opportunity to work, learn and study at the fantastic L3S Research Center. I have very much enjoyed the time at this great working environment and would like to thank everyone who contributed to this experience.

My journey at L3S began in the project ARCOMEM, where I worked primarily with Dr. Thomas Risse on the challenges around *named entity evolution*. This work already started during my Master's thesis under the supervision of Dr. Nina Thamasebi. In fact, this was the first time I got in touch with L3S and the exciting research they have been doing. Therefore, I would like to thank both of them for the nice and smooth start and the work we did together.

Later, when the ALEXANDRIA project started early 2014, my focus shifted towards the very interesting area of Web archives and related issues, which roused my passion. At this point, I started working primarily with Dr. Avishek Anand, who joined L3S at that time and is now a Professor at the Leibniz University of Hannover. He has been a great mentor and has supported my studies ever since to today. I would like to thank him a lot for all the fruitful discussions we had as well as all the studies we carried out together.

During this time, I also partially worked in another project, named FID Math, together with the German National Library of Science and Technology (TIB), primarily with Mila Runnwerth. Despite the limited time and resources, we developed many novel ideas and got so much work done, which deserves a big thanks as well.

Another special thanks goes to the Internet Archive for being a very valuable partner in ALEXANDRIA. Without the data they provided, much of my work would not have been possible. Especially, I would like to thank Jefferson Bailey and Vinay Goel from the Internet Archive, who have been amazing collaborators throughout the project.

Last, but not least, I would like to thank everyone, who is not mentioned here, but supported my work in one way or the other, including all colleagues at L3S as well as my family.

FOREWORD

Since early 2014, beginning with the inception of the EU project ALEXANDRIA[1], my research focus has been on the access and use of Web archives, which shaped and coined the entire course of my PhD studies. The work in this project involved extensive big data processing, data analysis as well as software development or tool building and required deep knowledge of related topics, such as information retrieval and data management.

Just like futurist and visionary Kevin Kelly (see quote after the foreword), I am convinced that in the future, the Web of the past will gain in importance for various areas and even integrate closer with the live Web. Through this work, I developed a strong interest in Web archiving and the archived Web as valuable dataset for scientists, but also for regular users to get a glimpse of the past.

Therefore, I have studied this interesting topic with different target groups in mind that have different interests and different perspectives on the use of Web archives. Based on these, the thesis has been structured into three views, each focusing on another aspect:

- the *user-centric view* in Chapter 2 deals with the needs of regular users and tools to make Web archives more accessible to them
- the concepts and tools presented in the *data-centric view* in Chapter 3 address data scientists as users, who want to study the archived Web
- the model of a graph as a way to approach Web archives in the *graph-centric view* in Chapter 4 is primarily of interest for researchers, but has also been shown to be useful in combination with the other views

This conception of different perspectives on Web archives was presented at the *Web Archiving Week 2017*[2] and published in the following paper, which provides the basis for my introduction in Chapter 1:

[1] Helge Holzmann and Thomas Risse. Accessing Web Archives from Different Perspectives with Potential Synergies. In *Researchers, Practitioners and Their Use of the Archived Web*, London, UK, jun 2017. School of Advanced Study, University of London. doi: 10.14296/resaw.0001. at the 2nd International Conference on Web Archives / Web Archiving Week (RESAW/IIPC)

The core contributions of this thesis in the individual chapters are published in the following articles:

[1] http://alexandria-project.eu

[2] http://netpreserve.org/wac2017

- The contributions in **Chapter 2**, which deals with the user-centric use of Web archives (**Browsing the Web of the Past**) are published in:

  – [2] Helge Holzmann and Avishek Anand. Tempas: Temporal Archive Search Based on Tags. In *Proceedings of the 25th International Conference Companion on World Wide Web - WWW '16 Companion*. ACM Press, 2016. doi: 10.1145/2872518.2890555

  – [3] Helge Holzmann, Mila Runnwerth, and Wolfram Sperber. Linking Mathematical Software in Web Archives. In *Mathematical Software – ICMS 2016*, pages 419–422. Springer International Publishing, 2016. doi: 10.1007/978-3-319-42432-3_52

  – [4] Helge Holzmann, Wolfgang Nejdl, and Avishek Anand. On the Applicability of Delicious for Temporal Search on Web Archives. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval - SIGIR '16*, Pisa, Italy, 2016. ACM Press. doi: 10.1145/2911451.2914724

  – [5]* Helge Holzmann, Wolfram Sperber, and Mila Runnwerth. Archiving Software Surrogates on the Web for Future Reference. In *Research and Advanced Technology for Digital Libraries, 20th International Conference on Theory and Practice of Digital Libraries, TPDL 2016, Hannover, Germany*, Hannover, Germany, 2016. doi: 10.1007/978-3-319-43997-6_17

  – [6] Helge Holzmann, Wolfgang Nejdl, and Avishek Anand. Exploring Web Archives through Temporal Anchor Texts. In *Proceedings of the 2017 ACM on Web Science Conference - WebSci '17*, Troy, New York, USA, 2017. ACM Press. doi: 10.1145/3091478.3091500

  – [7] Helge Holzmann and Mila Runnwerth. Micro Archives as Rich Digital Object Representations. In *Proceedings of the 10th ACM Conference on Web Science - WebSci '18*, Amsterdam, Netherlands, 2018. ACM Press. doi: 10.1145/3201064.3201110

- The contributions in **Chapter 3**, which deals with the data-centric use of Web archives (**Analyzing Archival Collections**) are published in:

  – [8] Helge Holzmann, Wolfgang Nejdl, and Avishek Anand. The Dawn of Today's Popular Domains - A Study of the Archived German Web Over 18 Years. In *Proceedings of the 16th ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '16*, pages 73–82, Newark, New Jersey, USA, 2016. IEEE, ACM Press. doi: 10.1145/2910896.2910901

  – [9]* Helge Holzmann, Vinay Goel, and Avishek Anand. Archivespark: Efficient Web Archive Access, Extraction and Derivation. In *Proceedings of the 16th ACM/IEEE-CS Joint Conference on*

Digital Libraries - JCDL '16, pages 83–92, New York, NY, USA, 2016. ACM. doi: 10.1145/2910896.2910902

  – [10] Helge Holzmann, Vinay Goel, and Emily Novak Gustainis. Universal Distant Reading through Metadata Proxies with Archivespark. In *2017 IEEE International Conference on Big Data (Big Data)*, Boston, MA, USA, dec 2017. IEEE. doi: 10.1109/bigdata.2017.8257958

- The contributions in **Chapter 4**, which deals with the graph-centric use of Web archives (**Exploring Web Archives Through Graphs**) are published in:

  – [6] Helge Holzmann, Wolfgang Nejdl, and Avishek Anand. Exploring Web Archives through Temporal Anchor Texts. In *Proceedings of the 2017 ACM on Web Science Conference - WebSci '17*, Troy, New York, USA, 2017. ACM Press. doi: 10.1145/3091478.3091500

  – [11]* Pavlos Fafalios, Helge Holzmann, Vaibhav Kasturia, and Wolfgang Nejdl. Building and Querying Semantic Layers for Web Archives. In *Proceedings of the 17th ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '17*. IEEE, jun 2017. doi: 10.1109/jcdl.2017.7991555

  – [12] Pavlos Fafalios, Helge Holzmann, Vaibhav Kasturia, and Wolfgang Nejdl. Building and querying semantic layers for web archives (extended version). *International Journal on Digital Libraries*, Jul 2018. doi: 10.1007/s00799-018-0251-0. URL `https://doi.org/10.1007/s00799-018-0251-0`

  – [13] Helge Holzmann, Avishek Anand, and Megha Khosla. What the HAK? Estimating Ranking Deviations in Incomplete Graphs. In *14th International Workshop on Mining and Learning with Graphs (MLG) - Co-located with 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, London, UK, 2018

  – [14] Helge Holzmann, Avishek Anand, and Megha Khosla. Delusive pagerank in incomplete graphs. In *Complex Networks and Their Applications VII*. Springer International Publishing, 2019. ISBN 978-3-030-05411-3

Before ALEXANDRIA, my studies were focused around another temporal topic that is closely related to the work on Web archives, namely the evolution of named entities, as part of the EU project ARCOMEM[1]. The contributions on this as well as a few other related works that I was involved in during the course of my PhD are published in the following articles:

*This paper was nominated for the Best Paper award or acknowledged as one of the best papers of the conference.

[1] http://www.arcomem.eu

- [15] Nina Tahmasebi, Gerhard Gossen, Nattiya Kanhabua, Helge Holzmann, and Thomas Risse. Neer: An Unsupervised Method for Named Entity Evolution Recognition. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 2553–2568, dec 2012

- [16] Helge Holzmann, Gerhard Gossen, and Nina Tahmasebi. Fokas: Formerly Known As - a Search Engine Incorporating Named Entity Evolution. In *Proceedings of the 24th International Conference on Computational Linguistics: Demonstration Papers (COLING 2012)*, pages 215–222, dec 2012

- [17] Helge Holzmann, Nina Tahmasebi, and Thomas Risse. BlogNEER: Applying Named Entity Evolution Recognition on the Blogosphere. In *3rd International Workshop on Semantic Digital Archives (SDA) - Co-located with 17th International Conference on Theory and Practice of Digital Libraries (TPDL)*, volume 1091, pages 28–39, Valletta, Malta, 2013

- [18] Helge Holzmann and Thomas Risse. Named Entity Evolution Analysis on Wikipedia. In *Proceedings of the 2014 ACM Conference on Web Science - WebSci '14*. ACM Press, 2014. doi: 10.1145/2615569.2615639

- [19] Elena Demidova, Nicola Barbieri, Stefan Dietze, Adam Funk, Helge Holzmann, Diana Maynard, Nikolaos Papailiou, Wim Peters, Thomas Risse, and Dimitris Spiliotopoulos. Analysing and Enriching Focused Semantic Web Archives for Parliament Applications. *Future Internet*, 6(3):433–456, jul 2014. doi: 10.3390/fi6030433

- [20] Helge Holzmann and Thomas Risse. Extraction of Evolution Descriptions from the Web. In *IEEE/ACM Joint Conference on Digital Libraries*, pages 413–414, London, UK, sep 2014. IEEE Press, IEEE. doi: 10.1109/jcdl.2014.6970201

- [21] Helge Holzmann and Thomas Risse. Insights into Entity Name Evolution on Wikipedia. In *Web Information Systems Engineering – WISE 2014*, pages 47–61, Thessaloniki, Greece, oct 2014. Springer International Publishing. doi: 10.1007/978-3-319-11746-1_4

- [22] Helge Holzmann, Nina Tahmasebi, and Thomas Risse. Named Entity Evolution Recognition on the Blogosphere. *International Journal on Digital Libraries*, 15(2-4):209–235, apr 2015. doi: 10.1007/s00799-014-0135-x

- [23] Tarcisio Souza, Elena Demidova, Thomas Risse, Helge Holzmann, Gerhard Gossen, and Julian Szymanski. Semantic URL Analytics to Support Efficient Annotation of Large Scale Web Archives. In

Semantic Keyword-based Search on Structured Data Sources, pages 153–166. Springer International Publishing, 2015. doi: 10.1007/978-3-319-27932-9_14

- [24] Anett Hoppe, Jascha Hagen, Helge Holzmann, Günter Kniesel, and Ralph Ewerth. An Analytics Tool for Exploring Scientific Software and Related Publications. In *Research and Advanced Technology for Digital Libraries, 22nd International Conference on Theory and Practice of Digital Libraries, TPDL 2018*, Porto, Portugal, 2018. doi: 10.1007/978-3-030-00066-0_27

It will also expand in time. Today's Web is remarkably ignorant of the past. [...] Viewing an earlier version of a typical website is not easy, but in 30 years we'll have time sliders enabling us to see any past version. Just as your phone's navigation directions through a city are improved by including previous days, weeks, and months of traffic patterns, so the Web of 2050 will be informed by the context of the past...

from the book **The Inevitable: Understanding the 12 Technological Forces That Will Shape Our Future** *(2017)*

by ***KEVIN KELLY***
(founding executive editor of the Wired magazine)

Contents

1 Introduction

A significant portion of the record of our society either exists exclusively on the Web today or has been moving to the Web. Consequently, there has been a surge of collection, curation and preservation efforts to archive the live and ephemeral Web. Web archiving initiatives such as the *Internet Archive*[1] and the *Internet Memory Foundation*[2] have been involved in periodically archiving websites for over 20 years with collection sizes amounting to several hundreds of terabytes. Additionally, a large number of libraries, universities, and cultural heritage organizations have Web archiving programs [25], with a 2011 survey reporting 42 different Web archiving initiatives across 26 countries [26].

By offering a unique possibility to look at past events and temporal evolutions, longitudinal collections present many opportunities for various kinds of historical analyses [27], cultural analyses and Culturomics [28], as well as analytics for computational journalism [29]. Hence, with greater availability of Web archives and increasing recognition of their importance, a growing number of historians, social and political scientists, and researchers from other disciplines see them as rich resources for their work [30].

However, as Web archives grow in scope and size, they in itself also present unique challenges in terms of usage, access and analysis that require novel, effective and efficient concepts and tools for researchers as well as for the average user. In the following, we tackle these from three different perspectives: the *user-centric view*, the *data-centric view* and the *graph-centric view*. One natural way of conceiving these views is as different zoom levels to look at the same archival collection, as illustrated in Figure 1.1, starting with the user-centric view that targets single documents to be examined by regular users. By zooming out to the data-centric view, one can scale the examination up to the whole archival collection or subsets of it. In contrast, the broadest zoom level, the data view does not focus on the individual documents but deals with the structures that span an archive.

Another way of conceiving the relations among the views is by considering their levels of abstraction. While the data-centric view is rather low level, closest to the

[1] http://www.archive.org

[2] http://www.internetmemory.org

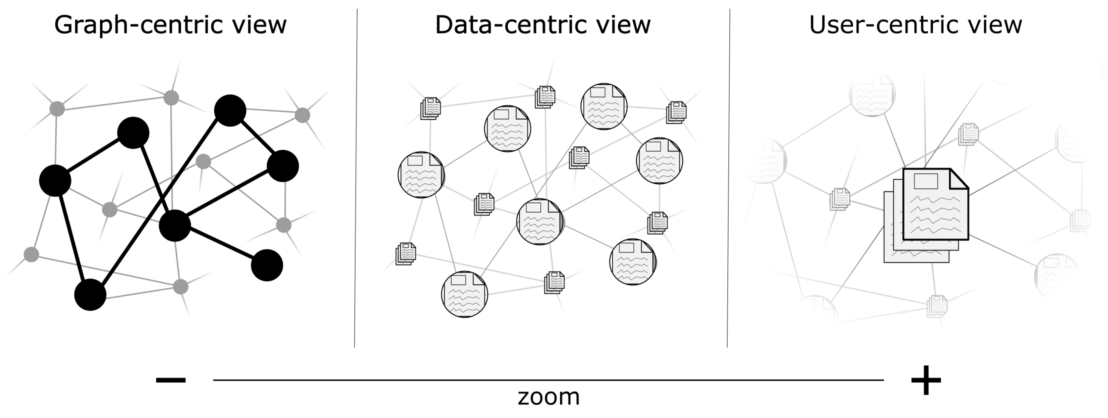


Figure 1.1. The three views on Web archives, representing different zoom levels to look at the archived data.

data as well as computational resources, the graph- as well as user-centric views can be considered more abstract. With the graph-centric view being a conceptual layer, not dealing with the technical details of data access and processing but the underlying conceptional models and relations, the user-centric view does not deal with such low-level issues at all but focuses on the users who interact with the archive without any particular skills required. This understanding leads to another distinguishable factor of the three views, namely the types of challenges they cause. While we care about usability as well as exploration in the user-centric view, technical and fundamental questions are raised to a much bigger extent by both other views. However, the difference among them becomes clear when we consider the problem that we touch upon in the very end in Section 4.2: the incompleteness of Web archives. From the data-centric view, this problem is not very obvious as we process the Web archive as given, tackling efficient access to the data and the analysis of what is in the studied collection. Only by zooming out and connecting contained concepts, issues like incompleteness are manifested, with the other side of a relation not being present. Finally, however, all three views are connected in one way or the other and there exist synergies among them in all the discussed ways to conceive the presented perspectives, as we will see later.

1.1 Browsing the Web of the Past

The natural way to look at a Web archive is through a Web browser, just like regular users explore the live Web as well. This is what we consider the **user-centric view**, as addressed in Chapter 2: access with a focus on users, their needs, and without requiring additional infrastructure or knowledge about the underlying data structures. Currently, the most common way to access a Web archive from a user's perspective is the *Wayback Machine*[3], the Internet Archive's replay tool to render archived webpages, as well as its open-source counterpart *OpenWayback*[4], which is available for most Web archives.

[3]http://web.archive.org
[4]https://github.com/iipc/openwayback

These tools are used by normal users who wants to look up an old webpage of the past as well as scholarly users who *closely* read individual webpages to understand their content and the context rather than or prior to zooming out and analyzing collections in a *data analysis* or *distant reading* fashion [31]. Similar to the live Web, where users either directly enter the URL of a webpage in a browser, click a link, or utilize search engines to find the desired page, the use of Web archives from a user's perspective can be distinguished into *direct access* and *search* as well.

User Access to Web archives

Direct access to an archived webpage through the Wayback Machine requires the user to enter a target URL first, before selecting the desired version of the corresponding webpage from a calendar view that gives an overview of all available snapshots of that URL. As URLs can be cumbersome, users on the live Web often prefer search engines over remembering and typing URLs manually. The Internet Archive's Wayback Machine provides search only in a very rudimentary way [32]. While their *Site Search* feature is a great improvement over plain URL lookups, their approach is pretty limited as it neither surfaces deep URLs to a specific page under a site nor supports temporal search, i.e., users cannot specify a time interval with their queries.

An alternative to search, if a URL is not known, is to follow hyperlinks from other pages. With Web archives being temporal collections, such a link needs to carry a timestamp in addition to the URL. Within the Wayback Machine, links are automatically temporal with timestamps as close as possible to the page or capture that is currently viewed. However, it is also possible to link from the outside of a Web archive, i.e., the live Web, to an archived page. In this case the timestamp needs to be set explicitly.

One way to do this is by manually pointing to a particular capture in a Web archive, like done in news articles about the case of Joy Reid, who claimed her blog was hacked and articles have been manipulated[5]. Another approach to form such temporal hyperlinks is by incorporating time information that can be associated to the link. We recently investigated this for the case of software that is cited or mentioned in scientific publications. We found that websites corresponding to software often nicely describe and document the referenced application and can be considered surrogates of the software's version that was referred to in an article [3, 5]. In this case, the publication date is a good indicator, or at least a close estimate, of the target time for linking the publication and mentioned software. While this example is very domain-specific to software, the same idea can be applied to other scenarios as well, such as preserving the evolution of people by archiving their blogs and social network profiles [33, 34, 35]. Another example is the preservation of Web citations, like on Wikipedia, to provide access to cited page at the time when it was cited[6].

Before we turn our attention to these direct access methods in Section 2.2

[5]http://ws-dl.blogspot.de/2018/04/2018-04-24-why-we-need-multiple-web.html
[6]https://en.wikipedia.org/wiki/Help:Using_the_Wayback_Machine

though, where we propose *Micro Archives* as rich digital object representations that can be referenced and linked [7], we will focus on *Web archive search* in detail.

Web Archive Search

Web archives can provide access to historical information that is absent on the current Web, like previous companies, products, events, entities etc. However, even after a long time of existence, Web archives are still lacking search capabilities that make them truly accessible and usable as temporal resources. *Web archive search* can be considered a special case of temporal information retrieval (temporal IR) [36]. This important subfield of IR has the goal to improve search effectiveness by exploiting temporal information in documents and queries [37, 38]. The temporal dimension leads to new challenges in query understanding [39], retrieval models [40, 41] as well as temporal indexing [42, 43]. However, most temporal indexing approaches treat documents as static texts with a certain validity, which does not account for the dynamics in Web archives where webpages change over time and hence, their relevance to a query may change over time. Furthermore, while information needs in IR are traditionally classified according to the taxonomy introduced by Broder [44], user intents are different for Web archives as studied by Costa and Silva [45]. In contrast to the majority of queries on the live Web being informational, queries in Web archives are predominantly navigational, because users often look for specific resources in a Web archive under a temporal aspect rather than general information that is in commonly available on the current Web as well. Costa et al. [46] presented a survey of existing Web archive search architectures and Hockx-Yu [30] identified 15 Web archives that feature full-text search capabilities. With the incorporation of live Web search engines, Kanhabua et al. [47] demonstrate how to search in a Web archive without indexing it.

In Section 2.1, we present a system with the goal to provide *temporal archive search*: given a keyword query together with a time interval we want to find the most authoritative pages, e.g., *"what were the most representative webpages of Barack Obama before he became president in 2005?"*. This would bring up Obama's senator website rather than his today's website and social media accounts. Such temporal semantics can often not be derived from the webpages under consideration and require external indicators. In our first version of Tempas, we incorporated tags attached to URLs on the social bookmarking platform Delicious as temporal cues [2]. Without evaluating the precision of the ranking, which was based on the frequency of a tag used with a URL, we show that this approach results in a good (temporal) recall with respect to query logs from AOL and MSN [4]. However, since Delicious is a closed system, available data is limited and our dataset only ranges from 2003 to 2011. Also, we found that it shows a strong bias towards certain topics, like technology. For these reasons, we switched to hyperlinks in the second version of Tempas. Using a graph-based query model, Tempas v2 exploits the number of websites and corresponding anchor texts linking to a URL in a given time interval. Its temporally sensitive search for authority pages of entities in Web archives has been shown to be very effective in multiple scenarios [6].

1.2 Analyzing Archival Collections

In contrast to accessing Web archives by closely reading pages like users do, archived contents can also be processed at scale, enabling evolution studies and big data analyses. In this **data-centric view**, addressed in Chapter 3, webpages are not necessarily considered self-contained units with a layout and embeds, but single resources are rather treated as raw data, such as text or images. A question like *"What persons appear together most frequently in a specific period of time?"* is only one example of what can be analyzed from the archived Web [48]. Typically, these studies do not require a whole archive though, but only on pages from a specific time period, certain data types or other facets that can be employed for pre-filtering the dataset. With ArchiveSpark we have developed a tool for building research corpora from Web archives that operates on standard formats and facilitates the process of filtering as well as data extraction and derivation at scale in a very efficient manner [9].

Web archives are commonly organized in two data formats: *WARC files* (Web Archive files) store the actual archived contents, while *CDX files* (Capture Index) are comprised of lightweight metadata records. The data-centric view approaches Web archives from these files, which is how data scientists would typically look at it. This perspective provides a higher, superior point of view, looking at whole collections rather than individual records, nicely rendered for a user. However, we have to deal with much lower data access and processing techniques at this level.

In the following, we distinguish between two perspectives in the data-centric view: 1. Web archives as the object of study, reflecting the *evolution of the Web and its dynamics*, 2. focusing on the contents of webpages to derive insights into the real world, referred to as *Web Science* [49].

Web Dynamics Analysis

Web archives that span multiple years constitute a valuable resource to study the evolution of the Web as well as its dynamics. In previous works on Web dynamics, suitable datasets had to be crawled first, which is tedious and can only be done for shorter periods [50, 51, 52, 53, 54]. With access to existing archives, more recent studies of the Web were conducted retrospectively on available data [55, 56, 57]. However, instead of analyzing the whole archive at once, all of them focus on a certain subset, such as national domains. Thanks to the Internet Archive we were provided with their entire subset of German pages over 18 years, i.e., the top-level domain `.de` from 1996 to 2013, which enabled us to carry out an analysis of the dawn of today's most popular German domains [8], presented in Section 3.1.

In this study, we explore how the age, volume and sizes of popular pages have evolved over the last decade. We found that most of the popular educational domains like universities have already existed for more than a decade, while domains relating to shopping and games have emerged steadily. Further, we observe that the Web is getting older, not in all its parts, but with many domains having a constant fraction of webpages that are more than five years old and aging further.

Finally, we see that popular websites have been growing exponentially after their inception, doubling in volume every two years, and also newborn pages have gotten bigger over time.

Web Archive Data Processing

Due to the sheer size of Web archives, in the order of multiple terabytes or even petabytes, distributed computing facilities are needed to process archived Web data efficiently. Common operations, like selection, filtering, transformation and aggregation, can be performed using the generic *MapReduce* programming model [58], as supported by *Apache Hadoop*[7] or *Apache Spark*[8] [59]. AlSum [60] presents with *ArcContent* a tool specifically for Web archives using the distributed database *Cassandra* [61]. In this approach, the records of interest are selected by means of the CDX records and inserted into the database to be queried through a Web service. The *Archives Unleashed Toolkit (AUT)*, formerly known as Warcbase, by Lin et al. [62] follows a similar approach based on *HBase*, a Hadoop-based distributed database system, which is an open-source implementation of Google's *Bigtable* [63]. While being very efficient for lookups, major drawbacks of these database solutions are the limited flexibility as well as the additional effort to insert the records, which is expensive both in terms of time and resources. In a later version, AUT/Warcbase allows the loading and processing of (WARC) files directly using Apache Spark in order to avoid the HBase overhead, for which they provide convenience functions to work with Web archives.

In contrast to that, we present in Section 3.2 a novel data processing approach for Web archives that exploits CDX metadata records for gains in efficiency while not having to rely on an external index [9]. ArchiveSpark is a tool for general Web archive access based on Spark. It supports arbitrary filtering and data derivation operations on archived data in an easy and efficient way. Starting from the small and lightweight metadata records it can run basic operations, such as filtering, grouping and sorting very efficiently, without touching the actual data payloads. In a step-wise approach the records are enriched with additional information by applying external modules that can be customized and shared among researches and tasks, even beyond Web archives [10]. In order to extract or derive information from archived resources, third-party tools can be integrated. Only at this point, ArchiveSpark seamlessly integrates the actual data for the records of interest stored in WARC files. Internally, ArchiveSpark documents the lineage of all derived and extracted information, which can serve as source for additional filtering and processing steps or stored in a convenient output format to be used as research corpus in further studies. Benchmarks show that ArchiveSpark is faster than competitors, like AUT/Warcbase and pure Spark in typical use case scenarios when working with Web archive data.

[7]https://hadoop.apache.org
[8]https://spark.apache.org

1.3 Exploring Web Archives Through Graphs

The final perspective, besides the *user-centric* and *data-centric views*, is addressed in Chapter 4, referred to as **graph-centric view**. This view enables the exploration of Web archives from a more structural perspective. In contrast to the views as discussed before, the focus here is not on content or individual archived records, but the relations among them. In the context of the Web, the most obvious relations are hyperlinks that connect webpages by pointing from one page to another. However, there is more that is less obvious. Looking at hyperlinks from a more coarse-grained perspective, multiple links can be combined to connections among hosts, domains or even top-level domains, revealing connections among services, organizations or the different national regions of the Web. Furthermore, by zooming out to the graph perspective after processing the archived data from a data-centric view, even relationships among persons or objects mentioned on the analyzed pages can be derived [48, 11, 12].

Similarly, the holistic view on archival collections provided by graphs is very helpful in many tasks and naturally generates synergies with the other views. The broad zoom level is crucial to get an overview of available records in an archive and to find the right resources. Hyperlinks among the archived pages can point users or algorithms in search or data analysis tasks to the desired entry points within the big and often chaotic Web archive collections. As shown before, we make use of this with our Web archive search engine Tempas (see Sec. 1.1). The effectiveness of hyperlinks and attached anchor texts for this task was already shown by previous works [64, 65, 66, 67].

Data Analysis

The mentioned approaches to explore Web archived through graphs, allow for queries on a structural level (cf. Fig. 1.1). Once a set of documents that match the query has been identified, a data-scientist may zoom in to look at the contents from a data-centric perspective. Quite commonly, such workflows also involve manual inspections of the records under consideration from a user-centric view. This is helpful to get an understanding of the data under consideration. Finally, derived results need to be aggregated and presented to the user in an appropriate form.

Figure 1.2 shows this generic analysis schema that outlines a systematic way to study Web archives. This schema can be adopted and implemented for different concrete scenarios. In such a setting, the graph-centric view is utilized to get an overview and find suitable entry points into the archive. This may initially be done manually by the user to get a feeling for the available data using a graph-based search engine like Tempas, but can also be integrated as the first step in a data processing pipeline to (semi-)automatically select the corpus for further steps. Next, the selected records can be accessed from a data-centric view at scale, using a tool like ArchiveSpark (see Sec. 1.2), to extract the desired information, compute metrics or aggregate statistics. Finally, the results are presented to the user. A concrete implementation of this pipeline is outlined in Section 4.1 (Sec. 4.1.4),

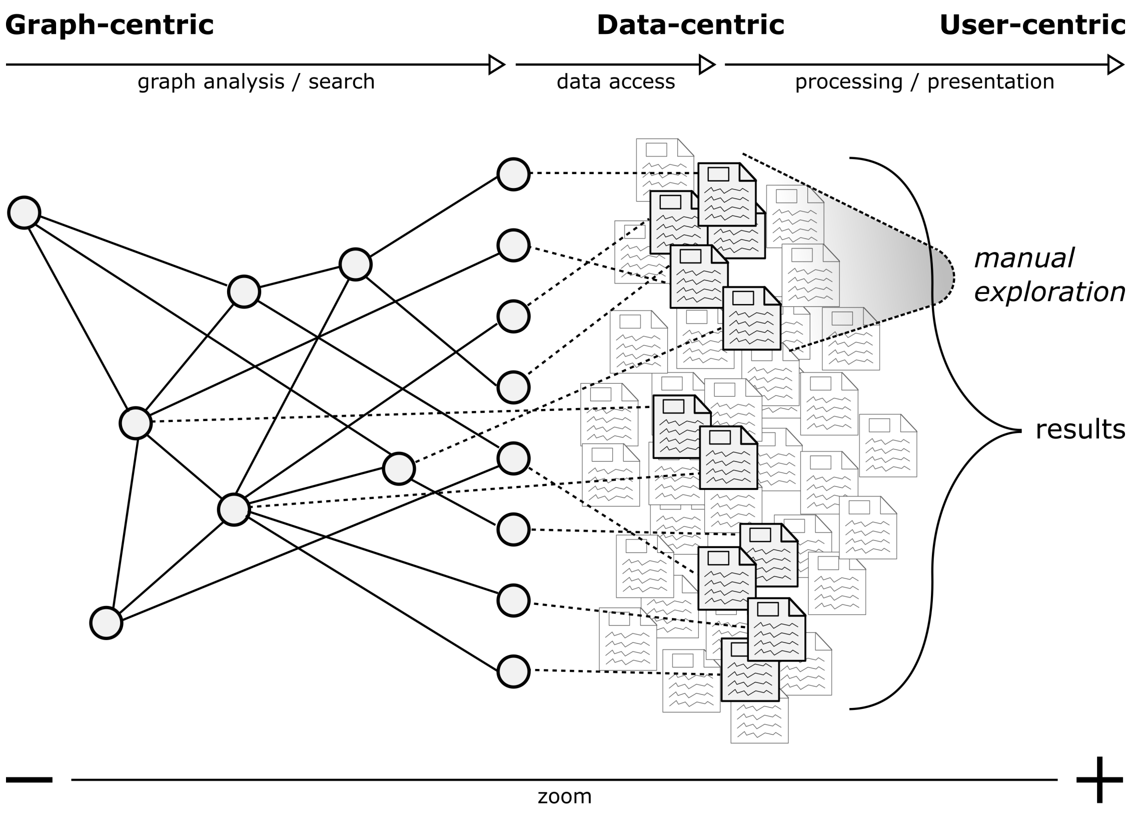


Figure 1.2. Combining different views on Web archives for systematic data analysis.

where we describe the example of analyzing restaurant menus and compare prices before and after the introduction of the Euro as Europe's new currency in Germany in 2001/2002.

Open Challenges

The reason for addressing the graph-centric in the very end, is because it requires a certain understanding of the other tasks to value its utility. While there are many synergies between graphs and the challenges discussed before, in which this structural perspective is very helpful, it also raises new issues and open questions. Graphs enable completely different kinds of analysis, such as centrality computations with algorithms like *PageRank* [68]. However, scientific contributions in this area specific to Web archives are very limited and results are less mature. Although scientists have looked into graph properties of the Web in general both on static [69, 70, 71, 72, 73] and evolving graph [74, 75, 76], we found that certain traits of Web archives lead to new kinds of questions. For instance, as we show in Section 4.2, the inherent incompleteness of Web archives can affect rankings produced by graph algorithms. Towards this, we present some early work on estimating this effect by proposing a measure based on the partial graphs extracted from different Web archives [13, 14].

2 User-centric View: Browsing the Web of the Past

The main difference of *Web archive search* as opposed to *live Web search* is its temporal dimension. This does not only lead to a different search behavior but also to different intents. Informational or transactional requests, in which the search engines act more like a question answering system or assistant for the user rather than a lookup system for webpages, can usually be served from the live Web. Even if the information or question that the user would like to get answered has a temporal aspect to it, like a historical event, it is quite likely that information about this are still documented somewhere on the live Web, e.g., in Wikipedia or more specialized information sources. Hence, users do not need to employ a Web archive for this. What they use Web archives for though, is to look up some old resource, whether it is some concrete URL or simply an abstract concept, such as a person's resume or some old news article, possible from a specific news source. These temporal navigational intents are what Web archive search should be able to answer.

After we discuss the concept of such a *Temporal Archive Search* system, Tempas for short, in Section 2.1, we will talk about methods to more directly integrate Web archives with the current Web as well as more traditional literature. While direct access is already well provided by the Wayback Machine, which allows for look-ups as well as temporal links with an explicit timestamp from any (live) webpage, we will look at less obvious cases, which demand for a temporal link, but in a more complex, often implicit, manner than pointing to archived version of a webpage. Such cases include references to inherently temporal objects or temporal states of evolving objects, which can be represented and documented by archived webpages. Examples, with a focus on scientific software, as well as novel concepts to create, share and link such *object representations* are presented and discussed in Section 2.2.

2.1 Temporal Archive Search (Tempas)

After the long-term preservation of the Web has been tackled as the first essential step in Web archiving by different organizations, the true potential of the archived collections can only be realized by enabling *effective search* and *exploration* over such collections. Unfortunately, search over Web archives has been very limited. Companies and organizations maintaining Web archives either provide only very rudimentary search interfaces or pure URL-lookup services like the Wayback Machine. Further, usage patterns on Web archives as a corpus of study are not very well understood, which results in a lack of training data for user intents and information needs. Due to the size of data in those archives and their temporal aspect, out-of-the-box search infrastructures with full-text indexes are resource and computationally expensive and largely do not fit the needs.

Whereas full-text search is beneficial for a wide variety of informational intents, there are specialized intents on archives for which we not always require indexing full-text. Specifically, since most of the intents for information in Web archives are navigational and temporal in nature, i.e. users are interested in specific resources and their evolution over time, full-text contents may not necessarily be useful here. Further, challenges like temporal ranking, link analysis and diversification are widely not solved yet [38]. Current retrieval models to rank versions of webpages are limited to relevance cues from document content [40]. This is primarily due to the inability of the models to determine which page was important at a given instant or interval of time. To make matters worse, it is even more difficult to identify the variations of a page that are the most interesting for users in a given time period only by analyzing *internal* properties of a page, like its content, as detailed in [38]. Hence, as there are often multiple versions of the same page, a big challenge is to identify which version is the most relevant with different textually relevant versions of the same page being relevant at different points in time.

While determining authority of pages in an archive independent of a query has been attempted by Nguyen et al. [77], popularity cues from external sources have not been considered. By incorporating *external* data, such as explicit temporal information about a website's popularity, this can be simplified and lead to a better retrieval performance. A source for that can be any dataset reporting about other websites, such as social network data, where users post their favorite or most controversial websites at a specific time of interest. Besides the explicit time information, another advantage of searching *external* data instead of websites itself is the more focused descriptions of only relevant pages. Users typically post the essence instead of the often verbose contents found on the websites, including layouts, comments, etc. Finally, this also allows for a leaner index, which is computationally less intensive for construction and storage as well as faster to query than a corresponding full-text index. These critical factors for ever growing Web archive collections with sizes in the order of hundreds of terabytes or even petabytes.

In view of these issues we propose an alternative search approach, which exploits external data sources as proxies for popular and historically relevant websites, instead of trying to compute those metrics on internal features of the archived websites. These surrogate information units have to be accurate enough to serve the targeted information needs and provide us with the required temporal relevance information. In the following we discuss two datasets as sources for this goal and evaluate their applicability for Web archive search by means of two corresponding versions of a prototype search system, called Tempas:

1. **v1**: built on data of the social bookmarking platform Delicious as an entirely external source, which contains descriptive pointers to webpages created by its users at different time points.

2. **v2**: incorporating hyperlinks and corresponding anchor texts, i.e., the clickable text of a hyperlink, from external webpages, extracted from the Web archive itself, that link and describe the destination pages in different time periods.

2.1.1 Related Work

Web archive search can be considered a special case of *Temporal Information Retrieval*. While information needs in Information Retrieval are traditionally classified according to the taxonomy introduced by Broder [44], user intents are different for Web archives as studied by Costa and Silva [45]. In contrast to the majority of queries being informational, where users search for information, in Web archives queries are predominantly navigational, because users often look for specific resources in a Web archive under a temporal aspect (cf. Sec. 2.1.2). The Internet Archive's Wayback Machine recently got a *site search* feature based on anchor texts [32], using an approach similar to ours. However, in contrast to Tempas system, the *Wayback Site Search* has no explicit temporal search support. Users cannot specify a time interval for their queries, and results are limited to homepages, i.e., the hostname of a URL without a path. Thus, it can find *Barack Obama*'s official website, but not his Wikipedia article or social media profiles.

Temporal Information Retrieval.

Temporal information retrieval has emerged as an important subfield in information retrieval with the goal to improve search effectiveness by exploiting temporal information in documents and queries [38]. The value of the temporal dimension was clearly identified by Alonso et al. [37] and has led to a plethora of work which utilizes temporal features in query understanding [39], retrieval models [40, 41] and temporal indexing [42, 43]. A survey by Campos et al. [38] gives an elaborate overview of the field. Most of the temporal retrieval models either focus on temporal informational intents [40] or are concerned with increasing recall with diversification [41, 78]. Temporal indexing approaches [42, 43] over Web archives assume documents to be versions of full-text content. A survey of existing Web archive search architectures was presented by Costa et al. [46]. We posit that building a suitable temporal full-text index for Web archive data is challenging and expensive though and has never been shown to be sufficiently effective. Contrary to previous approaches that concern themselves with full-text indexes and ad-hoc retrieval tasks we focus on building minimalistic indexes specifically for temporal navigational intents.

Effectiveness of Anchor Texts

Anchor texts are incorporated by Tempas v2 as a consequence of the limited and largely closed data from social media sources, such as Delicious, used in v1. The effectiveness of anchor texts for the task of *site finding* was already shown by Craswell et al. [64], though not in the context of Web archives or a temporal setting. They are reported to be twice as effective as searching the contents of pages. The authors in Kraaij et al. [65] combined anchor texts with content features for the task of *entry page search* and also found that search just based on anchor texts outperforms basic content features. In a similar experiment, Ogilvie and Callan [66] showed that anchor texts are the most effective features among others, such as full text and title, for the task of finding homepages and are only slightly behind full-text search for finding so-called *named pages*. Koolen and Kamps [67] re-evaluated the effectiveness of anchor texts in ad-hoc retrieval and showed that propagated anchor text outperforms full-text retrieval in terms of early precision on the TREC 2009 Web track. The authors in Kanhabua and Nejdl [79] studied anchor texts in a temporal context and analyzed their value in Wikipedia. Similar to our findings presented in Section 2.1.6, they were able to observe evolutions of entities through anchor texts, such as the transition of *Barack Obama* from senator to president. They also proposed a temporal anchor text model for their study, though specific to Wikipedia.

2.1.2 User Intents and Problem Statement

User intents formulated as queries and issued to a Web search engine are commonly classified by their information needs into *informational*, *navigational* and *transactional*. Broder [44] analyzed query logs and found that around a half of the queries are informational. The other half is roughly split into 40% navigational and 60% transactional queries.

These proportions are different for Web archives. There is seldom the need to issue an informational query to a Web archive, partly because most informational facts and intents can be served on the current Web as well. Also, transactional queries, which refer to an action that a user wants to perform, e.g., chat or shop online, are typically not applicable in an archive. Therefore, the majority of queries to a Web archive are navigational.

Costa and Silva [45] confirmed this assumption by analyzing query logs of their full-text search engine for the *Portuguese Web Archive*[1]. They report more than a half of the queries to be navigational. From the other half a large majority was

[1] http://arquivo.pt

informational with only 5-10% being transactional. However, what they consider transactional is much more specific than the original definition, such as downloading an old file or recovering a specific website. Similarly, their informational need refers to collecting information about a subject in the past and can often be interpreted as navigational.

Indeed, all information needs of Web archives could be considered navigational in a broad sense. That is, instead of navigating to a specific resource, we want to navigate to a specific information or subject, e.g., an entity. Some of these entities are represented on the Web by their personal or official websites, others by profiles on social networks or sub-pages on related websites, as well as Wikipedia or similar knowledge bases. We refer to these *central* resources as *authority pages* for a subject / an entity. These are dynamic though and may change over time by some disappearing or moving to a different domain as well as new ones emerging.

Objectives of Tempas

The main objective of Tempas is to meet the information need of a user exploring a Web archive and fulfill the user's intent as defined above: Given a textual keyword query together with a time interval we want to identify those webpages that are central for the subject addressed by the query in the specified time period. For instance, before the *European Union* received its own `.eu` top-level domain in 2005, the official website resided under `.eu.int`. Another kind of pages that are of interest when working with Web archives are those in a certain category or with a certain type of contents, such as online shops or restaurant menus. In contrast to queries for *authority pages*, which are rather precision oriented, here recall matters, for instance in data mining tasks (such a scenario will be discussed much later in Sec. 4.1.4).

In summary, both types of navigational queries serve as important entry points into huge Web archives, which is what we are aiming for. Even though users commonly have a subjective understanding of this problem, a quantitative evaluation is not trivial due to the lack of a crisp definition of an authority page or appropriate entry point. Moreover, we found that existing relevance judgments used in Web information retrieval are not suitable for evaluating this task. For instance, in the *TREC 2012 Web Track* ad-hoc judgements[2], `phoenix.edu` was considered irrelevant for the query *university of phoenix*, which we consider a perfect hit. Therefore, we conduct alternative evaluations by assessing the applicability of social bookmarks for Tempas in terms of its coverage and completeness in Section 2.1.4 as well as qualitatively evaluating the performance of anchor texts for Tempas based on example queries in Section 2.1.6.

Example Scenario

A typical scenario for Tempas is to find event related versions of an entity's websites in a Web archive. This can be important for researcher who want to study the topic

[2]`http://trec.nist.gov/data/web2012.html`

of an event from the past with the Web archive as their scholarly source. Today, the same websites might not be relevant for that topic anymore or do not even exist anymore. For that reason, the website would not be available in the index of a current search engine or cannot be discovered by a user using the same keywords. Consequently, it cannot be looked up in the Wayback Machine since that would require to know the exact URL of a desired resource.

An example of such a scenario is the election campaign website of Barack Obama for the US presidential election in 2008: *change.gov*. Today, the website shows an image stating the transition has ended and the new administration has begun[3]. Also, it is not among the top search results on Google anymore. A query for *obama election 2008* primarily yields more current websites reporting about the election. A researcher who is interested in reproducing the campaign might however be more interested in original content from that time. Also, regular users who just want to revisit the pre-election promises to compare with the achievement of the elected government would want to look at the original websites from back then.

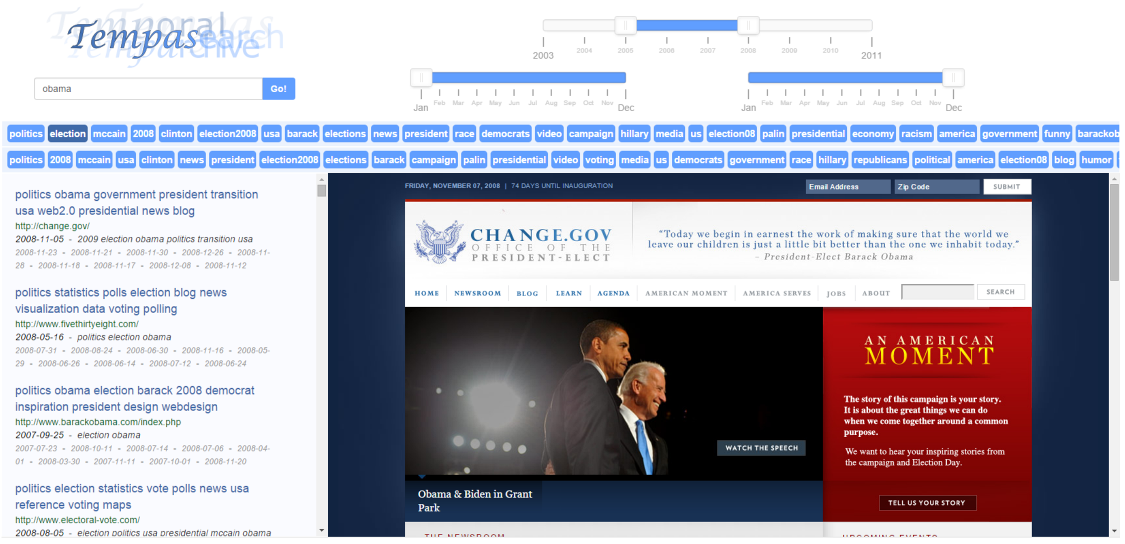
2.1.3 Tempas v1: Based on Social Bookmarks

With this version of Tempas we explore the idea of taking advantage of social media metadata about archived websites as cues for their importance. It is not uncommon for commercial search engines to cross-reference social media feedback in designing features for the same reason. One distinct advantage of such data is direct human endorsement of websites that they find interesting. A second and even more important aspect for the perspective of a temporal search engine is its temporal annotation. Metadata from external sources like social networks is typically timestamped, which proves to be a useful asset in identifying temporal importance of websites.

Delicious[4] as a formerly very popular social bookmarking platform is one of the social media services that works in this fashion. Users post popular links and describe them with a concise set of tags as succinct descriptors. Tags are single terms, for instance topics and subtopics, which together label and describe a website. In addition to that, these tags carry temporal information that can be exploited for search: While the tag *community* is frequently assigned to *facebook.com* today, other communities, such as *myspace.com* were tagged with the term before. The idea to base search on tags has been previously explored, but never in a temporal dimension [80, 81, 82]. We now present a version of our Web archive search engine Tempas that incorporates tags from Delicious in order to enable richer search capabilities on archived webpages than currently available. Tempas v1 is deployed under:

http://tempas.L3S.de/v1

[3]http://www.change.gov, visited: 22/12/2015
[4]https://del.icio.us


Tempas
obama
Go!
2003
2004
2005
2006
2007
2008
2009
2010
2011
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
politics election mccain 2008 clinton election2008 usa barack elections news president race democrats video campaign hillary media us election08 palin presidential economy racism america government funny barackob
politics 2008 mccain usa clinton news president election2008 elections barack campaign palin presidential video voting media us democrats government race hillary republicans political america election08 blog humor
FRIDAY, NOVEMBER 07, 2008 | 74 DAYS UNTIL INAUGURATION
Email Address
Zip Code
SUBMIT
CHANGE.GOV
OFFICE OF THE PRESIDENT-ELECT
"Today we begin in earnest the work of making sure that the world we leave our children is just a little bit better than the one we inhabit today." – President-Elect Barack Obama
HOME
NEWSROOM
BLOG
LEARN
AGENDA
AMERICAN MOMENT
AMERICA SERVES
JOBS
ABOUT
SEARCH
WATCH THE SPEECH
Obama & Biden in Grant Park
AN AMERICAN MOMENT
The story of this campaign is your story. It is about the great things we can do when we come together around a common purpose.
We want to hear your inspiring stories from the campaign and Election Day.
TELL US YOUR STORY

politics obama government president transition usa web2.0 presidential news blog
http://change.gov/
2008-11-05 - 2009 election obama politics transition usa
2008-11-23 - 2008-11-21 - 2008-11-30 - 2008-12-26 - 2008-11-28 - 2008-11-18 - 2008-11-17 - 2008-12-08 - 2008-11-12

politics statistics polls election blog news visualization data voting polling
http://www.fivethirtyeight.com/
2008-05-16 - politics election obama
2008-07-31 - 2008-08-24 - 2008-06-30 - 2008-11-16 - 2008-05-29 - 2008-06-26 - 2008-06-14 - 2008-07-12 - 2008-06-24

politics obama election barack 2008 democrat inspiration president design webdesign
http://www.barackobama.com/index.php
2007-09-25 - election obama
2007-07-23 - 2008-10-11 - 2008-07-14 - 2008-07-06 - 2008-04-01 - 2008-03-30 - 2007-11-11 - 2007-10-01 - 2008-11-20

politics election statistics vote polls news usa reference voting maps
http://www.electoral-vote.com/
2008-08-05 - election politics usa presidential mccain obama

Figure 2.1. Tempas showing search results for *obama* and *election* between Jan 2005 and Dec 2008.

Overview

The initial prototype of Tempas is designed as shown in Figure 2.1. Relating to the example from Section 2.1.2, it shows a query for *obama* during the time when he was senator of Illinois from 2005 to 2008 before he became president. The first suggestion bar, right below the query input, lists those terms that were most relevant to the query during the selected time, which are frequently co-occurring tags of the issued query. Of course, one of the top ranked tags here is the election, which can be selected to refine the search results and focus on this particular sub-topic. This opens up a second suggestion bar which is slightly more aligned towards the election and re-ranks the tags according to their co-occurrence with both query tags *obama* and *election* during the selected time period. The results shown in the left panel are those websites which were most relevant for the users of Delicious with respect to the given query terms and selected time period. Besides Barack Obama's official website and a website on statistics about the election on the second and third rank, the first rank is actually his election campaign website *change.gov* as desired.

To get an impression what is behind those websites, similar to search results on Google or other search engines, every result includes a title. On Tempas v1 this comprises the most related tags during the time period of the query. For the desired election campaign website, these tags describe it as a political website of Obama and his government, which includes news and blog articles. This description does not necessarily correspond to the content of the websites, but instead represents a temporal view on the websites by its visitors.

Up to this point, all information has been compiled purely based our external source Delicious, without deriving data from the actual Web archive or computing a ranking function on internal characteristics. Neither is it required to have the entire archive on-site. Only if the user clicks on a result it opens a version from the queried time period using the Internet Archive's Wayback Machine or any other Memento compliant Web archive[5].

By that, Tempas serves as an effective entry point to Web archives and naturally provides high accurate results with respect to the underlying external resource. Researchers using these results in their studies should be aware of the bias introduced by the dataset, however, it allows them to build a corpus for their research which is well-defined and easily comprehensible. While more advanced search and ranking methods are often complex and their performance is questionable, especially on temporal datasets such as Web archives, the results on Tempas solely correspond to their temporal popularity on the external data source. Besides ranking up the most temporally as well as topically related websites, like *change.gov* in our example, it also filters the vast amount of noise and low-quality websites on the Web, which are not included or less frequent in the external data.

Dataset

This work is based on the data of Delicious from 2003 to 2011, collected by Zubiaga et al. [83]. The dataset, called *SocialBM0311*, has been published online and is freely available[6]. It contains the complete bookmarking activity for almost 2 million users from the launch of the social bookmarking website in 2003 to the end of March 2011 with 339,897,227 bookmarks, 118,520,382 unique URLs, 14,723,731 unique tags and 1,951,207 users. Its size is 11GB of compressed, tab-separated text data with each line in the following form:

```
<url_md5 user_id url unix_timestamp tags>
```

In the following we will refer to a URL as website or use the terms interchangeably. Every record in the dataset with its specific time is referred to as a version of the website. In the final system this is linked to a capture in the archive, which is a snapshot of when the website was crawled.

Data and Query Model

We operate on the tag dataset described above where websites, considered as our documents $d \in \mathcal{D}$, are tagged with labels $l \subseteq \mathcal{L}$ at a given time $t \in \mathcal{T}$. We allow for a discrete representation of time and assume a granularity of days. Each tuple in the dataset can be represented as a triple $(d, l, t) \in \mathcal{D} \times 2^{\mathcal{L}} \times T$. Note that such a tuple represents the version of the document d at the time t. A temporal query $q = (q_l\,, q_t)$ has a text component $q_l \in \mathcal{L}$ and a time period of focus $q_t \in \mathcal{T} \times \mathcal{T}$. We require that the results for the temporal selection induced by the query return versions of the documents which are valid in q_t. In what follows, we use the terms websites and documents interchangeably.

[5]http://mementoweb.org

[6]http://www.zubiaga.org/datasets/socialbm0311

Ranking Documents and Tags

For designing the retrieval model, we take the following desiderata into consideration:

1. Most relevant websites in a given time interval with respect to certain query tags are also most frequently tagged with the query terms during this time frame.
2. More relevant versions of a website in a given time interval with respect to a set of query tags are tagged with more of these tags and less other tags.
3. Most frequently co-occurring tags of given query tags in a certain time interval represent their most related tags/topics during this time frame.

First, we retrieve a set of relevant documents $R(q)$ which are valid in q_t. A document is considered relevant if its versions in q_t cover the query terms q_l. In other words, the union of tags of all versions of $d \in R(q)$ in q_t must cover q_l.

For ranking, we follow a nested ranking approach in which we first rank documents or websites and then rank its corresponding versions. Based on our desiderata, we compute the score of each document as the product of the *mutual information* of the document $d \in R(q)$ and the query terms in q_t along with the popularity of the document. The popularity of the document d is measured by the frequency of versions of d tagged in q_t. Note that there could be multiple tuples for the same document with or without the same tag sets. Next, following the second desiderata, we rank the versions for a given document based on vanilla counts of query tags associated with each version.

Finally, since we are also interested in retrieving related tags, we also retrieve a set of relevant co-occurring tags given q_l. A tag is deemed relevant if it co-occurs with the query terms in q_l. Similar to the document relevance, a tag might be relevant even if it does not co-occur with all tags in q_l for a version of $d \in R(q)$ if it co-occurs with the remainder of the tags is some other version of d. The tags are scored and aggregated across all documents based on weighted counts of their co-occurrences to give a final ranked list of most relevant co-occurring tags.

Index Structures

The core of Tempas v1 is a collection of indexes and mappings, which are tailored for retrieving the above described result sets. All of them are built to provide retrieval with a monthly granularity. We created indexes to retrieve tags as well as websites based on a query consisting of tags q_l for a time period q_t (i.e., *TagTagMapping*, *TagUrlMapping*). Furthermore, we created a year and monthly based index to retrieve tags without providing tags as input (i.e., *YearTagMapping*, *MonthTagMapping*) for exploratory search for a particular time interval without providing tags as input. Another index allows retrieving all versions of a website that have been tagged during a given time period together with the tags (i.e., *UrlTagMapping*). For a compact index structure, two mappings assign ids to tags and websites, which are used exclusively in all other indexes and mappings (i.e., *IdTagMapping*, *IdUrlMapping*). Even though there is much room for improvement and optimization of temporal indexes [43, 84], the rather simple mappings, which can easily be constructed using a distributed data processing platform like *Hadoop*, already go a long way.

All indexes are ordered by their keys (i.e., ID, year, month or pair of tag and year or website and year, respectively). The fetching of the indexes is realized by Web services, which are invoked separately for tags and websites as well as a single fetch for the versions with tags of each website. After fetching the data, all items (i.e., tags, websites or versions) are ordered according the described ranking functions.

The query with tags as well as the time period, consisting of start year and month as well as end year and month, can be entered and selected at the top of the page, using the search input and the three sliders as shown in Figure 2.1.

2.1.4 Evaluation of Social Bookmarks for Tempas

In the following we analyze the applicability of Delicious for searching webpages from the past, as employed by Tempas v1, based on the following research questions, with a focus on recall:

(**R1**) *What fraction of websites clicked by users on a search engine in the past are included in Delicious as well?*
(**R2**) *What topics or entities are covered by Delicious?*
(**R3**) *What are the natural limitations of Delicious as a dataset for temporal Web archive search?*
(**R4**) *How do posting times on Delicious and query times in search engines relate to each other?*

The more disputable precision or temporal relevance, which is difficult to evaluate as alluded to in Section 2.1.2, is broadly subjective and inherently given by Delicious when defined as popularity, considered as the number of users who bookmarked a certain page. For the later version of Tempas, based on anchor texts, this is qualitatively evaluated and discussed in Section 2.1.6.

Analysis

The Delicious dataset spans nine years from 2003 to 2011 and contains about 340 million bookmarks, 119 million unique URLs, 15 million tags and 2 million users [83]. Each bookmarked URL is timestamped and tagged with descriptors. The methodology of our analysis is shown in Figure 2.2.

We begin with a query workload and identify the associated tags for each query in Delicious. This is done by using Bing search results as a proxy and selecting tags attached to the returned URLs. We then compute the overlap of clicked search results in two query logs, from AOL and MSN, and the URLs tagged with the query or its expanded tags to compute the *recall* of Delicious for temporal

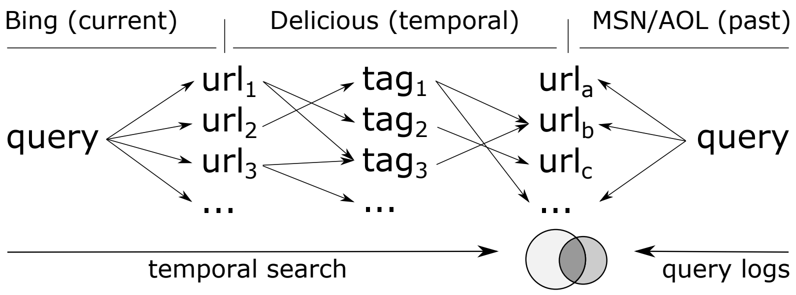


Figure 2.2. The analysis workflow from query-tag mapping using Bing and Delicious to querying Delicious and comparing against MSN/AOL query logs.

search. This is done for overlapping time intervals of both datasets, query logs and Delicious, which are May 2006 for MSN and March to May 2006 for AOL.

As an example scenario, consider the clothing brand `American Apparel`. Using the approach as described below, we map this query to the tags `americanapparel` and `apparel` as well as `t-shirts`, which appears to be used quite synonymously on Delicious. In the next step, we retrieve all URLs that were tagged with any of these tags during a given time, here May 2006. This results in 227 URLs. As a ground truth we compare against query logs from MSN and AOL from the same time. These contain two and four URLs that users clicked on for the query `american apparel`: `americanapparel.net` and `americanapparelstore.com` on MSN as well as `allonlinecoupons.com` and `usawear.org` additionally on AOL. As only the first two are contained in the set of URLs from Delicious, it reaches a full recall of 100% (1.0) with respect to MSN and 50% (0.5) w.r.t. AOL. Analyzing the precision remains for future work as it will require an appropriate retrieval model to rank the relevant links up to the top. A high recall, however, is a crucial prerequisite to make the system usable in practice.

Query-Tag Mapping. Our query workload comprises article titles from the German Wikipedia, which we consider as entities in the following. In principle, the approach is applicable to any type of queries, but the mapping of entity names to tags is more straightforward than arbitrary multi-keyword queries, which would usually refer to a set of tags instead of single representatives. The used Wikipedia collection consists of 1.8 million articles from the main namespace without disambiguation and list pages. The focus on German aligns with the Web archive available to us and also narrows down the vast number of articles on Wikipedia. The titles were issued as queries to Bing between August 7 and 13, 2015 and we stored the top 100 results for each query.

To map the queries to tags that best represent the corresponding entity, we collected all tags from Delicious that were attached to the retrieved URLs from Bing and used by at least 10 users. From these we kept only those tags that were used by at least 10% of the users who posted one of the URLs for that query. In addition, we selected a reference tag w_{ref} tag that exactly matches the query after down-casing

Entity	Tags	#Q	Recall
ESPN	`espn`	7492	0.60
Gmail	`gmail`	5285	0.92
Craigslist	`classifieds, popular, imported, craigslist`	4943	0.80
Wikipedia	`wiki, encyclopedia, wikipedia`	3063	0.60
Imdb	`cinema, imdb`	2626	0.92
AIM	`aim`	2180	0.57
YouTube	`flv, converter, youtube`	1965	0.75
Sudoku	`sudoku`	1739	0.75
PayPal	`paypal`	1710	0.60

Table 2.1. Top 10 entities according to MSN query logs from May 2006 with a recall value of greater than 0.5.

and removing special characters. E.g., $w_{\text{ref}}(\text{Barack Obama}) =$ `barackobama`. For the remaining tags we computed an adaption of *IDF* (inverse document frequency) based on the total number of considered queries and relative to the reference tag for normalization:

$$\text{idf}(w) = \frac{\log(|\text{queries}|)}{|\{q \in \text{queries} \mid w \in \text{tags}(q)\}|}$$

$$\text{rel.idf}(w) = \frac{\text{idf}(w)}{\text{idf}(w_{\text{ref}})}$$

This number indicates the generality of the tag, i.e., how specific it is to the assigned query. Additionally, we computed what we call *exclusiveness*. A high exclusiveness indicates that the tag does not often co-occur with the reference tag, which would be uncommon if both tags represent the same entity:

$$\text{excl}(w) = 1 - \frac{\#\text{posts}(w, w_{\text{ref}})}{\min(\#\text{posts}(w),\ \#\text{posts}(w_{\text{ref}}))}$$

$\#\text{posts}(w_1, w_2, ...)$ defines the number of posts on Delicious with all specified tags $w_1, w_2, ...$ as unique pairs of user and posted URL to filter spammers with a large number of posts of the same URL.

To combine the numbers, we computed the average score of each tag w as $0.5 \cdot (\text{rel.idf}(w) + \text{excl}(w))$ for every query with reference tag w_{ref}. The resulting score indicates how specific a tag is to its considered query and at the same time how complementary or exchangeable it is to the reference tag, which we consider a representation of the corresponding entity. Experiments have shown that a threshold of 0.7 reliably identifies true representative tags for a query. Thus, we took all tags with a score equal or greater as well as w_{ref}.

Examples are shown in Table 2.1. Although this simple approach is not accurate in all cases, it is good enough for an experiment like ours. While it worked well for entities like `Wikipedia`, it sometimes yields too general tags, like `popular` in case of `Craigslist`. For other entities, some tags seem too generic at the first glance, but appear to be used almost synonymously on Delicious, such as the format `flv` for `YouTube`. However, the quality is acceptable and a better mapping would have only resulted in even higher recall values as we will show in the following.

Querying. To assess the recall of the considered Delicious dataset, we *queried* it for the identified tags and at the same time selected records for the corresponding entity from the MSN and AOL query logs. We matched queries with exact name as extracted from the Wikipedia titles, ignoring case. The clicked URLs for a given query served as ground truth in the experiment and we measured recall by counting how many of those could be also found in Delicious. Each URL annotated with one of the considered tags was selected. We operated under the assumption that tags identified by us were also used back in the past, which we believe is valid, as most tags relate to entity names or variations. In total, 12,106 entities could be successfully mapped to tags as well as corresponding queries in the MSN query logs and 12,170 in the AOL logs.

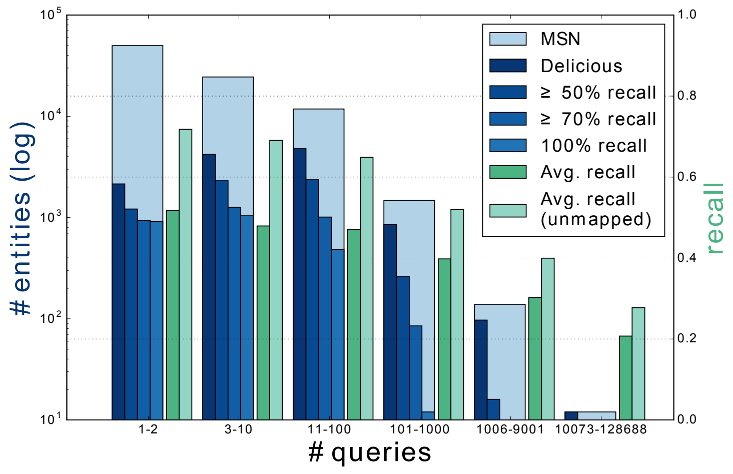


Figure 2.3. Comparison of the numbers of entities in the MSN query log vs. Delicious with different recalls and the average recall considering entity-tag mapping or not, partitioned by the popularity of entities according to number of queries in the logs.

From Table 2.1, which shows the top 10 entities with the highest frequency in the MSN logs and a recall of greater than 0.5 on Delicious, a certain bias towards rather technical and Internet related entities can be observed (**R2**). A bias to a certain, less popular type of entities could also be affirmed by counting numbers and recall values for entities of different popularities as shown in Figure 2.3: Even though all of the most popular entities could be mapped to tags in Delicious, their recall is relatively low. However, the highest presence in Delicious was not among the least popular entities either, but among those that were queried by up to 100 search sessions in the MSN logs. Also, the highest recall values were reached for the long-tail as well, rather than for the more popular entities (**R3**).

To ensure the observed recalls are not due to a poor mapping (see *Query-Tag Mapping* above), we computed the recall values of the same entities also by considering the entire Delicious dataset, not only results retrieved by the mapped tags (*unmapped*). This suggests the upper bound, which could potentially be reached given a better entity-tag mapping. However, as presented in the figure, these recall values are only slightly higher than the ones achieved with our mapping.

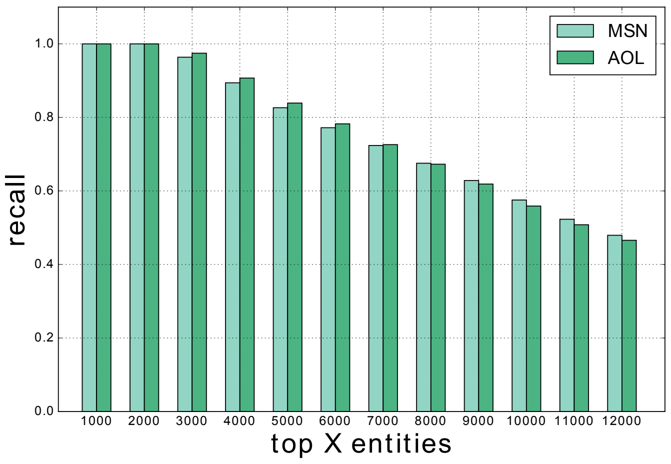


Figure 2.4. Top X entities according to their Delicious recall with respect to MSN and AOL query logs.

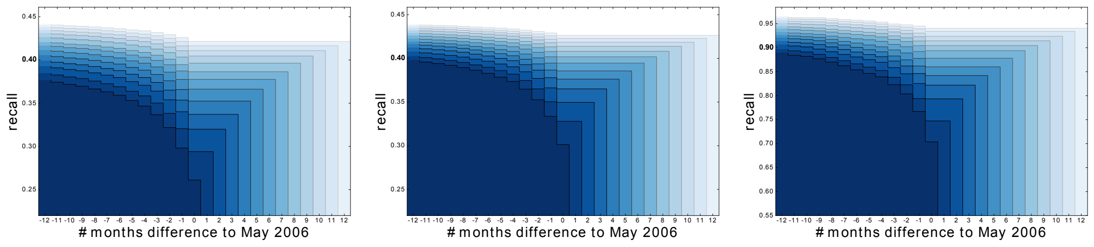
Overall, the average recall values of the entire experiment resulted in 48% w.r.t. MSN and 46% w.r.t. AOL. Although this is already a good result, it gets even better by looking at those entities, which are strongly represented on Delicious and reached the highest recalls in the experiment. As presented in Figure 2.4, for the top 2000 entities we achieve a full recall w.r.t. both query logs. The top 6000, which is about 50%, still exhibit a recall of almost 0.8 with even slightly higher values w.r.t. AOL (**R1**).

This shows, about half of the URLs that users clicked in a search engine can be found by using Delicious as an external data source to search Web archives. Even more intriguing, if used with the queries that are best represented by the dataset, we can even reach much higher values up to a recall of 100%.

Temporal Search

So far, none of the presented analyses has taken into account the temporal aspect, although our ultimate goal was to evaluate the applicability of Delicious as external data source for temporal Web archive search. Thus, we wanted to know, given the fairly good recalls achieved before, are the URLs spread across the entire dataset or focused around the query times of the available logs. Figure 2.5 presents these temporal recall results. To get comparable values, we queried Delicious with the same approach as before just only around the time spans of the logs, i.e., May 2006 for MSN (0) and March (-2) to May 2006 (0) for AOL. The x-axis in the figures denotes the time difference in months up to one year in the future and the past.

Notable is that in all plots, up to a difference of around three months, the recall grows faster by relaxing the search interval to the past. However, from the third month on the recall gain is higher by expanding the search interval to the future. This suggests posts on Delicious are lagging slightly behind the queries in the considered search engines. Overall the results are a slightly higher w.r.t. the AOL query logs, which, however, may be due to the fact that it spans three months instead of one, covered by MSN. Hence, in May Delicious is already two months ahead of AOL. This corresponds to the observation that the recall w.r.t. MSN for three months from 0 to 2 approximately matches the recall w.r.t. AOL in month 0. Overall, the recall results come very close to what we observed in Figure 2.4. While we achieved a full recall for the top 2,000 entities from the entire dataset,



(a) all entities (MSN) **(b)** all entities (AOL) **(c)** top 2000 entities (AOL)

Figure 2.5. Tempas v1 recall w.r.t. MSN query logs from May 2006 (0) and AOL query logs from March 2006 (-2) to May 2006 (0).

already a search span of two years, 12 months in the future and 12 in the past, is sufficient to retrieve 95% w.r.t. MSN and even more than 95% w.r.t. AOL, as presented in Figure 2.5c. Even more intriguing, we only lose about 10% of recall by searching as little as two months in the past and three months in the future (**R4**). However, in favor of Delicious was the fact that the platform was relatively popular during the analyzed time. Thus, the results might be lower today and call for alternative datasets.

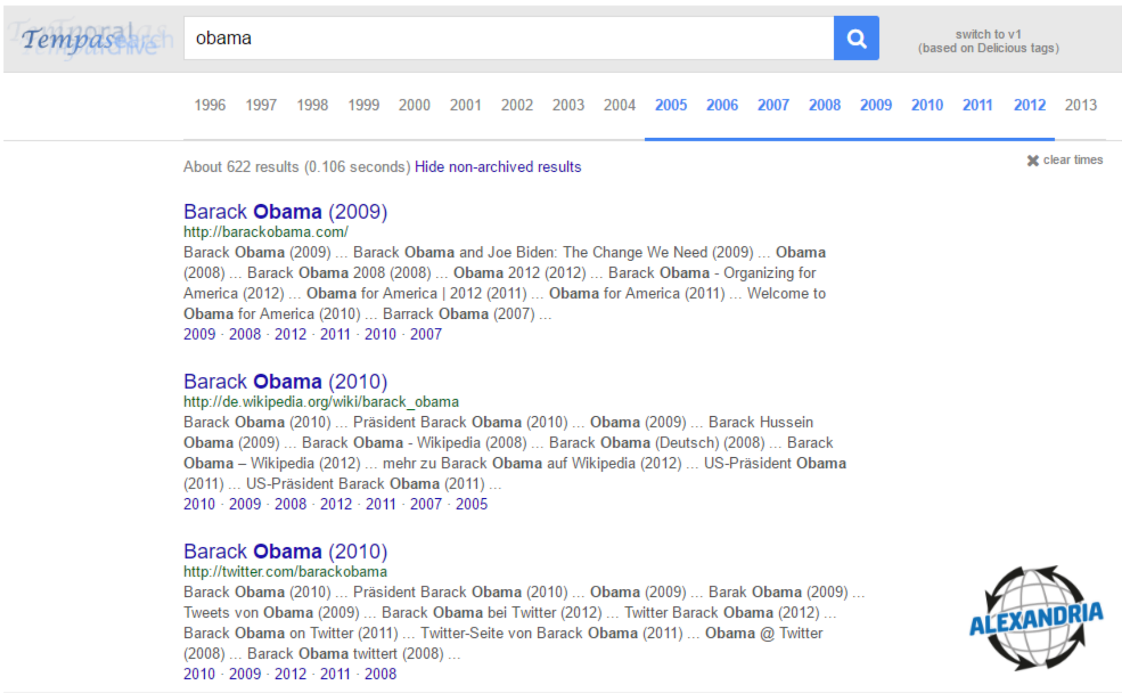
2.1.5 Tempas v2: Based on Anchor Texts

Switching to anchor texts with the second version of Tempas is the natural consequence to make the approach of incorporating external cues for temporal search more sustainable, since Delicious is a closed system and the available data is limited (cf. Sec. 2.1.3). We also showed that it is very biased towards a certain group of users and the shift to anchor texts can deliver better results, for more diverse queries. Tempas v2 is based on anchor texts extracted from a temporal Web archive corpus. We built it on the observation that anchor texts are crucial text segments, which in many cases succeed well to describe succinctly the target webpage, and hence are a natural choice for navigational queries.

Instead of the Delicious tags as used in Tempas v1, we now identify temporal anchor texts as surrogate information units of the target webpages in a given time period and propose lean index organization methods to support temporal, navigational needs. Anchor texts are short, concise, important and non-noisy descriptors of information content, typically desired by navigational queries. Just like the tags before, anchor texts as surrogates are many orders smaller than the original full text of a page, resulting in a leaner index.

With Tempas v2 we propose a temporal retrieval model based on such anchor texts that ranks webpages not only by textual relevance but under consideration of their decisive times. While this method does not cover pure informational needs, it successfully identifies results beyond navigational needs including representative pages for entities that vary over time. With this model we present a fully functional indexing and retrieval system under:

http://tempas.L3S.de/v2


obama

switch to v1
(based on Delicious tags)

1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013

About 622 results (0.106 seconds) Hide non-archived results

✖ clear times

Barack **Obama** (2009)
http://barackobama.com/
Barack **Obama** (2009) ... Barack **Obama** and Joe Biden: The Change We Need (2009) ... **Obama** (2008) ... Barack **Obama** 2008 (2008) ... **Obama** 2012 (2012) ... Barack **Obama** - Organizing for America (2012) ... **Obama** for America | 2012 (2011) ... **Obama** for America (2011) ... Welcome to **Obama** for America (2010) ... Barrack **Obama** (2007) ...
2009 · 2008 · 2012 · 2011 · 2010 · 2007

Barack **Obama** (2010)
http://de.wikipedia.org/wiki/barack_obama
Barack **Obama** (2010) ... Präsident Barack **Obama** (2010) ... **Obama** (2009) ... Barack Hussein **Obama** (2009) ... Barack **Obama** - Wikipedia (2008) ... Barack **Obama** (Deutsch) (2008) ... Barack **Obama** – Wikipedia (2012) ... mehr zu Barack **Obama** auf Wikipedia (2012) ... US-Präsident **Obama** (2011) ... US-Präsident Barack **Obama** (2011) ...
2010 · 2009 · 2008 · 2012 · 2011 · 2007 · 2005

Barack **Obama** (2010)
http://twitter.com/barackobama
Barack **Obama** (2010) ... Präsident Barack **Obama** (2010) ... **Obama** (2009) ... Barak **Obama** (2009) ... Tweets von **Obama** (2009) ... Barack **Obama** bei Twitter (2012) ... Twitter Barack **Obama** (2012) ... Barack **Obama** on Twitter (2011) ... Twitter-Seite von Barack **Obama** (2011) ... **Obama** @ Twitter (2008) ... Barack **Obama** twittert (2008) ...
2010 · 2009 · 2012 · 2011 · 2008

Figure 2.6. Screenshot of Tempas v2 for query '*obama*' in period 2005 to 2012.

Overview

The previous design of Tempas was driven by the structure of its underlying dataset as well as the desired temporal search capabilities. In contrast to that, Tempas v2 has been designed closer to the layout of a common search engine for the current Web, in order to make it more accessible for regular users, but also, with the temporal features of v1 in mind. As a result, it is all temporal again while resembling the look of a familiar search engine.

Users can formulate their information need by specifying a textual query with the option of selecting a time interval of interest. The screenshot in Figure 2.6 shows the graphical interface. After a query is issued (here: *obama*) the user can select a time interval for the search in yearly granularity, as opposed to the monthly granularity supported by the first version, which was shown in Section 2.1.4 to be unnecessarily narrow in order to achieve a reasonable recall. The results are presented as shown in the screenshot with titles, snippets and years compiled of matched anchor texts and sorted according to their temporal popularity based on usage frequencies. The exact model used will be elaborated further in the following passages.

Searching Anchor Texts

Anchor texts are a special type of text and should not be treated as running text, like articles or similar content. We are purely using texts included in a link, i.e., the text a user can click on to follow the link. Although those snippets have some disadvantages in being potentially less descriptive than the text surrounding a link or the content of a linked page, they have great advantages for our primary use case of finding *authority pages* as described above:

- Anchor texts describe with a high confidence the linked webpage with **little distracting or unrelated content**. This prevents pages from showing up for unrelated query terms and gives a higher relative relevance to page actually related to a query term.
- Often, anchor texts contain the name of the linked page or a **concise label of the linked content**. Hence, these terms are frequently used to link to authority pages of entities. For instance, the name of a person is outstandingly often used to link to the person's website, which is therefore likely to be ranked high for this query.
- As probably intended by the original reason for hyperlinks on the Web, anchor texts frequently point to pages containing a more detailed description of the contained terms. This way, instead of repeating **descriptions or definitions**, pages link to the most meaningful explanation of the linked text, which often is a social profile, the official website, or an encyclopedic article, e.g., *Wikipedia*.
- Within a website, anchor texts are used as **navigational elements** to refer to certain parts of the site, e.g., the menu on a restaurant site. The same term is then commonly used in connection with the site's name to deep link into that part of the website from external pages.

Temporal Retrieval Model

The retrieval model for Tempas v2 is based on the given Web archive as well as a fixed, pre-defined temporal granularity (we use one year). We split the time period covered by the provided Web archive collection into equally sized time intervals of this granularity $\{[t_0, t_1], [t_2, t_3], \ldots, [t_{n-1}, t_n]\}$. For each of these time intervals we create a temporal Web representation, derived from a set of links $L_{[t_a,t_b]}$. In the presented implementation of Tempas, $L_{[t_a,t_b]}$ comprises only those links that appeared in the given data during the corresponding interval and has not existed or discovered before, assuming that it was created in that time period, later referred to as $L_{\texttt{emergence}}$ later. This conception of links that are *posted* on the Web with their *emergence* indicating some kind of growing attention of the linked target page, is derived from the respective temporal graph model G. More details on that as well as different graph models, based on snapshots representing a specific state of the Web or with all links merged across a time period will be presented and discussed later in Chapter 4 (*Graph-centric View*). With the graph, the set of links, as well as corresponding anchor texts A, we now define a Web model for each time interval under consideration:

$$\mathcal{W}_{[t_a,t_b]} = (G_{[t_a,t_b]}, A_{[t_a,t_b]}, L_{[t_a,t_b]})$$

In the following we omit the interval and treat the Web model $\mathcal{W} = (G, A, L)$ for every interval independently. From the graph $G = (V, E)$ of W, each node $v \in V$ represents a page that is either part of the Web archive in the current interval or is linked to from such a page but not necessarily contained itself. Let $freq(v, a)$ be a scoring function used to compute the relevance of the page represented by node v for a given anchor text $a \in A$. We define this function based on the edges $e = (u, v) \in E$ with source $u \in V$ and destination v for which a link $l \in L$ exists with anchor text a (l_{uva}). Instead of counting these edges directly, we count the number of different hosts $host(u)$ of the source nodes u of the edges, i.e., the hostname in the URL of the page corresponding to node u, e.g., `en.wikipedia.org` in `https://en.wikipedia.org/wiki/World_Wide_Web`:

$$freq(v, a) = |\{host(u) | e = (u, v) \in E \wedge (e, a) \in L\}|$$

Host frequencies have turned out to be more resistant against link spam in our experimentation. While many pages linking to a single URL may all belong to the same website and hence, created by the same domain owner, these are counted only once in our system. To compute the **relevance score** $rel(v, a)$ this frequency score is normalized based on the maximum among all $v \in V$ and all $a \in A$, which results in numbers between 0 and 1. Finally, we introduce a multiplicative factor γ to get positive scores along with a logarithmic function to dampen the differences:

$$rel(v, a) = log(\frac{freq(v, a)}{\max_{v \in V, a \in A} freq(v, a)} \cdot \gamma)$$

This score is used to **boost the textual relevance** of the query, i.e., a double boost means a match is twice as important. The textual relevance is computed among all anchor texts for a page that fall into the same relevance class $\varphi = relc(v, a)$, which we consider the floor of the relevance score, i.e., the greatest integer less than or equal to this score:

$$relc(v, a) = \lfloor rel(v, a) \rfloor$$

For the boosting we multiply the relevance class score with the logarithm of the maximum score to account for time intervals with low overall frequencies. Since it is *easier* for a page to receive a high relevance score if only very few pages are archived in that time period and potentially link to a page, we want to reward those hits for a query that receive a high relevance score in a time period with a larger number of pages in the archive:

$$boost(\varphi) = \varphi \cdot log(\max_{v \in V, a \in A} freq(v, a))$$

The **result ranking** is then computed based on the textual relevance scores retrieved from our index (see below) for all relevance classes φ and boosted according to the boosting score as defined above. The final score is the linear combination of all boosted scores.

Year	Count	Year	Count	Year	Count
1996	17	2002	3109	2008	28764
1997	166	2003	16222	2009	30132
1998	92	2004	225066	2010	49658
1999	120	2005	42378	2011	64692
2000	128	2006	85691	2012	87444
2001	955	2007	111561	2013	48871

Table 2.2. Maximum frequencies in Tempas v2 per year.

Index Construction and Retrieval

The indexes for our Tempas v2 system have been computed according to the models described above on the German Web archive from 1996 to 2013, which was collected by the Internet Archive. It consists of more than 2 billion distinct archived webpages under the German `.de` top-level domain. These link to a total of 26,443,384,902 URLs, not only under `.de`. After filtering malformed / invalid URLs as well as those that are very infrequently linked, i.e., $rel(v, a)$ is smaller than 1 for any anchor text a that links to the page represented by node v, we are left with 319,574,156 URLs that go into our index. The maximum frequencies of links from distinct hosts to distinct destinations with distinct anchor texts, that are used for normalizing the relevance scores as well as for boosting, are listed in Table 2.2.

Tempas v2 is implemented using *Elastic Search*[7] (ES). ES creates a separate **full-text index** for each indexed field in its **schema**. We defined this schema such that a field of a document, i.e., a page / URL, represents a relevance class φ in one time interval. We used a yearly granularity for building our indexes, so one time interval represents a year $y \in \{1996, 1997, \ldots, 2013\}$: $[t_a, t_b] = [\text{y/01/01 - 00:00:00}, \text{y/12/31 - 12:59:59}]$. For each of these time intervals we extracted the link list $L_{\texttt{emergence}}$ and corresponding temporal model (see Sec. 4.1). To compute the relevance scores for our retrieval model as described above, we set the parameter $\gamma = 10,000$, i.e., we consider four decimal places of the normalized frequencies, and used basis 3 for the logarithm. This results in relevance classes between 0 and 8. Finally, the anchor texts $a \in A$ that describe any of the links to a URL represented by a node $v \in V$ form the document of the corresponding webpage, with a indexed in the field of its relevance class $relc(v, a)$, e.g.:

```
{
  "url": "http://.../wiki/World_Wide_Web",
  "years": {
    "2013": {
      "8": ["World Wide Web", "WWW", ...],
      "7": ["internet", ...]
```

[7] http://www.elastic.co

```
            ...
         },
         "2012": {
            ...
         },
         ...
      }
   }
```

Depending on the selected time period the corresponding fields are queried. Textual relevance is computed by ES among all anchor texts in these field based on a vector space model using a variant of `tf-idf`, i.e., a combination of the *term frequency*, *inverse document frequency* and *field-length norm*[8]. This is boosted and averaged as defined above and the results are ranked accordingly. The title and snippets shown in Figure 2.6 are generated from matching anchor texts sorted by the boost values of the fields that the anchor text appears in as returned by ES's *highlight* feature. The same order is used for the years listed below each search result, with the first year being selected as the main year linked by the title.

2.1.6 Evaluation of Anchor Texts for Tempas

In the following we give a qualitative evaluation of the Tempas v2 system. Before we will look at some example queries and analyze the results returned by Tempas for these queries in more detail, we discuss our general observations of the system.

General Discussion

In our current Tempas version not all the returned search results are available in the underlying Web archive, as that is not checked when the indices are built. Instead, we provide a feature to check search results for presence in the Internet Archive's Wayback Machine at the displayed years after they have been returned to the user and hide them in case they are not archived. In the following we will ignore this and discuss all results returned by Tempas, regardless of whether they are archived or not.

Anchor text search: Anchor texts have special characteristics and should not be treated as running texts as we discussed in Section 2.1.5. We found our result rankings often to be highly satisfactory for queries taking those characteristics into account, but less useful when formulated differently. For instance, the name of a website typically yields what is expected, while the topic does not. E.g., the query *google* results in `google.com` and `google.de` at the top ranks, however, the query *search engine* does not even return these URLs on the first page. A reason for this is that famous websites are usually linked by their name instead of a description,

[8]https://www.elastic.co/guide/en/elasticsearch/guide/current/scoring-theory.html

while the descriptive terms may be part of another site's name, which is then ranked higher, such as `searchenginewatch.com`.

This meets our objective of finding suitable entry points into a Web archive, such as a specific page even if the **URL changes over time**. However, we do not cover the opposite case where the name of a website has changed, and possibly the URL too. For such renamings one would like to find the former URL under the current name, which would require a deeper evolution analysis of the Web graph, which is out of the scope of this work.

Another assumption is that pages are sufficiently frequently linked to. This is not always the case especially in the earlier periods of our Web archive due to the limited number of available pages (cf. Table 2.2). As a result, the performance of Tempas is better for queries at query intervals starting from around the mid 2000s.

German Web dataset: Although the index of Tempas was built from a German Web archive, it is not limited to webpages from the German Web but can also find pages under different top-level domains that are linked from a page under `.de`. For multi-lingual websites, such as *Wikipedia*, the German versions are typically preferred, as these are more frequently linked from other German websites and therefore receive higher frequency scores (see *Temporal Retrieval Model* in Sec. 2.1.5).

We observed that English query terms or non-German entities work quite well in many cases and return the expected results, but overall result in fewer hits, e.g., *obama* has only 724 results, while *merkel* has 11,913. This is usually not critical, as similar to what happens with popular search engines like Google or Bing, often only the very first hits in Tempas are relevant for a query. While that is typically the first page, i.e., first ten hits, in Tempas we found that very often only the first one to five results are subjectively very relevant to the query. This can be partially explained by the diversification features of the big, multi-purpose search engines, as well as their goal to meet various kinds of information needs as opposed to the more focused navigational needs we are addressing. Thus, a general query issued to search engines is often multi-faceted and the sought information is scattered among multiple pages, while navigational queries on Google or Bing are usually answered by only a few hits.

Temporal granularity: By temporally searching Tempas, i.e. entering both a textual query and a time interval, we found that the quality of results is much lower when only single years are selected as opposed to selecting a range of consecutive years. Even though our indices are built on a yearly basis (with separate fields for each relevance class), it appears that only combining multiple indices leads to the expected results. By further analyzing this issue, we found that more famous pages are permanently linked over time but often do not show peaks for single years like less popular pages sometimes do. Averaging over multiple years results in a smoothing of these peaks and drops the subjectively less relevant results below these temporally highly frequent hits. For that reason, all example queries discussed below are issues for periods of multiple years instead of single years.

***obama* @ [2005, 2006]**
1. `http://obama.senate.gov`
2. `http://de.wikipedia.org/wiki/barack_obama`
***obama* @ [2005, 2007]**
1. `http://myspace.com/barackobama`
4. `http://obama.senate.gov`
5. `http://youtube.com/profile?user=barackobamadotcom`
***obama* @ [2008, 2013]**
1. `http://barackobama.com`
2. `http://de.wikipedia.org/wiki/barack_obama`
3. `http://twitter.com/barackobama`

Table 2.3. Selected temporal hits for query '*obama*'.

Example Queries

Let us now discuss a few example queries that we consider to be potentially interesting for the problem of temporally navigational queries in Web archives: *Barack Obama*, *Angela Merkel*, *European Union*, *Creative Common License* and *Wikipedia*. All of them feature temporal characteristics, stressing different aspects.

Barack Obama. One of those entities popular all around the world is the US president. During the later times of our dataset this was *Barack Obama*. Therefore, he will serve as our first example query. Figure 2.6 shows the results for the time from when he became Senator of Illinois in 2005 until 2012 when he was re-elected as president. Although in this case the query is only his last name *obama*, we receive hits solely for *Barack Obama* as he is more prominently linked on the German Web than for example his wife *Michelle Obama* and search result diversification features are not implemented in our retrieval model.

When searching this long time frame of eight years, Tempas finds the overall most prominent authority websites of Barack Obama in these years, as expected: 1. his official website, 2. his Wikipedia article, 3. his Twitter account. More temporally sensitive results are retrieved when meaningful time frames of Barack Obama are queried, as shown in Table 2.3. For instance, in 2005 / 2006, i.e., Obama's first two years as senator of Illinois, his senate page is the top hit, followed by his Wikipedia article. By extending the time interval to 2007, that page gets pushed to rank 4, caused by the rise of social media with his *Myspace* page taking the lead and his *YouTube* profile on rank 5. In between are German news articles reporting about his run for president (not shown in Table 2.3). Starting from when Obama was elected president in 2008 we get the same results as discussed above, including his official website and Twitter replacing Myspace as his main social media profile. Before 2005 there are no hits for Barack Obama at all, because he was very famous in Germany and therefore, not sufficiently linked.

Angela Merkel. An equally famous politician, especially in Germany, is the

***merkel* @ [2000, 2004]**
1. `http://merkel.de` *(university bookstore Merkel)*
2. `http://angela-merkel.de`
***angela merkel* @ [2000, 2004]**
1. `http://angela-merkel.de`
2. `http://cdu.de/idx-merkel.htm`
3. `http://cdu.de/ueber-uns/buvo/pv/pv.htm`
***merkel* @ [2005, 2010]**
1. `http://angela-merkel.de`
2. `http://de.wikipedia.org/wiki/angela_merkel`
***merkel* @ [2010, 2013]**
1. `http://angela-merkel.de`
2. `http://facebook.com/angelamerkel?...`
3. `http://de.wikipedia.org/wiki/angela_merkel`
4. `http://twitter.com/search?q=%23merkel`

Table 2.4. Selected temporal hits for query '*merkel*' and '*angela merkel*'.

German chancellor *Angela Merkel*. However, it is interesting to search only her last name before 2005. In contrast to *Obama*, which always referred to Barack Obama and was not relevant at all before, *Merkel* was the name of a university bookstore, which received even more links from 2000 to 2004 than the later chancellor, as shown in Table 2.4. Unfortunately, this website is not present in the archive and the domain is used differently today.

Therefore, to query for the person Angela Merkel, her full name should be used in earlier years. After 2004 this does not make a difference anymore. Angela Merkel exhibits a similar evolution as the US president, which can be observed through Tempas as well. Before she was elected chancellor in 2005 she became the leader of her party *CDU* in 2000. Therefore, next to her official website, pages on the party's site are among the top hits. From 2005, with her election, also her Wikipedia page become more popular. Later, in 2010, her social media profiles on Facebook and Twitter began to gain popularity.

European Union. Like many international organizations, the European Union's official website was located under the top-level domain `.int`: `http://europa.eu.int`. In 2005, they received their own top-level domain `.eu`: `http://europa.eu`. Today the `.int` URL does not exist anymore and the `.eu` one replaced it completely. This evolution is reflected by the queries shown in Table 2.5. In addition, as for most famous entities, their Wikipedia article has become one the most important resources about the EU.

This is a classic example where looking up a website in a Web archive is difficult as we need to be aware of the former URL that was active at the time of interest. Without a system like Tempas the best bet would be the current URL of the EU, which did not exist prior to 2005.

***european union* @ [1996, 2005]**
1. `http://europa.eu.int`
***european union* @ [2005, 2013]**
1. `http://en.wikipedia.org/wiki/european_union`
2. `http://europa.eu`
3. `http://europa.eu.int`

Table 2.5. Selected temporal hits for query '*european union*'.

Creative Commons License. *Creative Commons* (CC) is one the most popular copyright and open content licenses. Since the inception of the CC organization in 2001 and the release of the first version in 2002, there have been three updates until its current version 4.0 was released in 2013[9].

The different variants of the CC license, e.g., *BY-NC-SA*, *BY-NC-ND*, ..., are used by many projects on the Web. As they are commonly linked under the name *Creative Commons License*, their version history can be traced through the Tempas search results, shown in Table 2.6. At any time, the query leads to the current version of the license. Even though the URLs change over time, the query together with a timespan can be considered a temporal reference.

Wikipedia. Today Wikipedia is widely known under its domain `wikipedia.org` or corresponding language versions, e.g., `de.wikipedia.org` for the German version of Wikipedia. However, when it was launched in 2001, its domain was under `.com` and moved to `.org` one year later. Today, the `.com` domain and sub-domains forward to their `.org` counterparts and no one is aware of the old URLs anymore. Without this information, it is impossible to look up the early website of Wikipedia in a Web archive. Again, this is revealed by the search results in Tempas, shown in Table 2.7, which make such a lookup very easy.

[9] `https://wiki.creativecommons.org/wiki/License_Versions`

***creative commons license* @ [2002, 2003]**
1. `http://creativecommons.org/licenses/by-nc-sa/1.0`
2. `http://creativecommons.org/licenses/by-nd-nc/1.0`
***creative commons license* @ [2004, 2006]**
1. `http://creativecommons.org/licenses/by-nc-sa/2.0`
2. `http://creativecommons.org/licenses/by-nc-nd/2.0`
***creative commons license* @ [2007, 2013]**
1. `http://creativecommons.org/licenses/by/2.5`
2. `http://creativecommons.org/licenses/by/3.0`
3. `http://creativecommons.org/licenses/by-nc-sa/3.0`

Table 2.6. Selected temporal hits for query '*creative commons license*'.

***wikipedia* @ [2001, 2002]**
1. `http://de.wikipedia.com/wiki.cgi?wikipedia_willkommen`
2. `http://wikipedia.com`
3. `http://de.wikipedia.com`
***wikipedia* @ [2003, 2013]**
1. `http://de.wikipedia.org`
2. `http://de.wikipedia.org/wiki/hauptseite`
3. `http://wikipedia.de`
3. `http://wikipedia.org`

Table 2.7. Selected temporal hits for query '*wikipedia*'.

2.1.7 Conclusion and Outlook

Web archives are large longitudinal collections that store webpages from the past, which might be missing on the current live Web. Consequently, temporal search over such collections is essential for finding prominent missing webpages. However, this has been challenging due to the lack of popularity information and a proper ground truth to evaluate temporal retrieval models. Limited search and access patterns over Web archives have been well documented. One of the key reasons is the lack of understanding of the user access patterns over such collections, which in turn is attributed to the lack of effective search interfaces. Current search interfaces for Web archives are (a) either purely navigational or (b) have sub-optimal search experience due to ineffective retrieval models or query modeling. We identify that external longitudinal resources, such as social bookmarking data and anchor texts, are crucial sources to identify important and popular websites in the past. To this extent we present Tempas, a temporal search engine for Web archives. Tempas operates as a fairly non-invasive indexing framework, constituting an attractive and low-overhead approach for quick access into Web archives by not dealing with the actual contents. The ability to specify a time period together with the textual query enables temporal information retrieval capabilities for Web archives.

Websites are posted at specific times of interest on several external platforms, such as bookmarking sites like Delicious. The timestamped bookmarks on Delicious provide explicit cues about popular time periods in the past along with relevant descriptors. Attached tags do not only act as relevant descriptors, useful for retrieval, but also encode the time of relevance. These are valuable to identify important documents in the past for a given temporal query. With Tempas v1 we tackle the challenge of temporally searching a Web archive by indexing tags and time. We allow temporal selections for search terms, rank documents based on their popularity and also provide meaningful query recommendations by exploiting tag-tag and tag-document co-occurrence statistics in arbitrary time windows. Our evaluation shows, it is crucial to keep in mind the bias of any dataset used with the presented approach, as obviously only queries supported by the dataset can lead to satisfying results. It turns out, the best represented entities among the most popular ones on

Delicious are from the technical domain. Nevertheless, even overall we reached a promising recall of 46% to 48% w.r.t. the analyzed query logs from MSN and AOL. By focusing on the top queries, which exhibit the best recall results, a full recall of 100% could be retained up to the top 2000 of the about 12,100 analyzed entities. The top 50%, approx. 6000 entities, still result in a recall of almost 80% on average. Further, we investigated temporal recall values with respect to the time spans of the query logs around May 2006. It has shown that most of the websites of interest w.r.t. the query logs also appeared on Delicious around one year before and after the query times. Already five months around the query time result only in a little loss of recall.

In order to cover the missing entities over time for a more complete Web archive search, we investigated different sources for the same purpose and found anchor texts as meaningful text surrogates that can act as reasonable entry points for the exploration exploring of archived pages. This finding lead to Tempas v2, a new approach to searching Web archives based on temporal link graphs and corresponding anchor texts. By providing us with richer texts, anchor texts are also less biased as well as a less limited good, since they can be extracted from the Web archive itself. Departing from traditional informational intents, we show how temporal anchor texts can be effective in answering queries beyond purely navigational intents, like finding the most central webpages of an entity in a given time period. We propose indexing methods and a temporal retrieval model based on anchor texts. Finally, we discussed several interesting search results, which reflect the evolutions of entities but also provided useful entry points into the massive archives. In the remainder of the work, we will show the versatility of this in different use case scenarios, like a data analysis experiment in Chapter 4, where we showcase how temporal Web graphs, as exploited by Tempas v2, help to identify starting points for an analysis of a massive archive by using a tool like ArchiveSpark, which will be presented in Chapter 3.

2.2 Temporal References and Links

In the area of digital libraries and in the scholarly domain in general exist many digital identifiers used to reference objects and entities in literature, most prominently, the *Digital Object Identifier* (DOI) [85]. These identifiers are commonly backed by a set of metadata that describe the referenced object. While meta information are easy to create and maintain for fixed objects, such as scientific publications, which do not change anymore after they have been published and assigned their DOI, this approach does not scale well for more dynamic entities. Therefore, we now propose archived snapshots from the Web as an alternative as well as temporal representation for dynamically evolving entities.

As one such subject, we consider software, an omnipresent good in science that is often referenced in literature. Software is constantly being developed and can have a different state in every moment, especially if it is open source and being developed by a large community. In such cases, it is difficult to permanently keep corresponding metadata up to date. Even more challenging, a software that is developed by thousands of developers, with every developer working on a small piece of it, is nearly impossible to be precisely expressed by a fixed set of metadata values. Further is such a representation in many cases not what a reader requires to fully understand the referenced asset. Way more useful would be a description, documentation, or even the source code in case of software. We found that most of this information already exists on the Web [5], as we see in Section 2.2.2.

From an author's perspective who wants to reference some entity or object that is not explicitly prepared for this, the collection of all required meta information to comprehensively describe the referenced asset means a big additional effort. Instead, we often see very vague references in literature, e.g., only a name, sometimes with the version or date. Similarly, references to Web resources, such as blog articles, are made as a footnote containing the URL. However, even if the date of visit is specified, this is not very helpful as the referenced blog post or linked resources may already have changed by the time it is read.

Many of these problems could be solved if we had richer presentations of the cited objects. If the reader does not only see the name, version and author of a referenced software, but can actually read the documentation at the time when the author accessed it. For that reason, we propose *Micro Archives*: microscopic collections of archived resources on the Web that describe a single entity or object, cohesively preserved for future reference. While existing Web archives already provide the necessary infrastructures to preserve all required resources individually, Micro Archives can be considered a logical and semantic connection of such resources to provide a holistic view onto a cited object. Furthermore, metadata that may be available in unstructured or semi-structured form as part of such a Micro Archive can be dynamically extracted and presented as needed whenever required.

Before we get to the details of Micro Archives, we will first discuss a case study on the representation of scientific software on the Web and its coverage in Web archives to motivate the ideas and show off current problems as well as the potential of the presented approach. Finally, we present Micrawler, a modular proof-of-concept prototype that implements the entire pipeline of creating, archiving, analyzing, presenting and citing Micro Archives, along with a practical example of how our approach can be used within the scientific publication workflow. Further, we showcase two use case scenarios, i.e., 1) blog articles, 2) software, which we have investigated in terms of inconsistencies that could be fixed with Micrawler in the future. Finally, we will highlight the opportunities created by Micro Archives in various areas and stress why we think the presented concepts are an inevitable step in our digital world.

2.2.1 Related Work

Piwowar et al. [86] provided evidence that enhanced access to research data lead to an increased number of citations. Although there has been a quite some work on research data and its use in literature [87, 88, 89] as well as on Web archives as containers for cultural, personal or scientific entities [90, 33, 91], there is not much on combining both aspects as we intent with our work. Dynamic research data, such as software, has been neglected for a long time because of its volatility and its development process that cannot be suitably mapped by traditional metadata. Only recently, several initiatives have emerged to foster the use of software in a scientifically sound manner, such as the *Software Sustainability Institute*, *Software Heritage* or *FORCE11*[10] [92, 93, 94, 95]. However, we are the first to propose the incorporation of Web archives for this purpose.

Web Archiving

With the growing interest of Web archives, it has become a dire need to preserve scientific information before it vanishes from the Web [96, 97, 98]. Web archives have been gaining growing popularity as scholarly source [30] in disciplines like the humanities [99]. Further, Web archives have been of interest as subject of research themselves. In 2011, Ainsworth et al. [96] analyzed how much of the Web is archived and found that for up to 90% of the pages in the considered collections at least one archived version exists, however, only a few of them have a consistent coverage over time. Beside the question of how much is archived, we are particularly interested in what has been archived, which falls in the area of profiling Web archive collections [100, 101, 102]. Finally, the goal of this study is to investigate the applicability of Web archiving to preserve representative surrogates of entities. Even though this particular subject has not been tackled before, other researchers looked into Web archiving to create preservation copies of other types or Web resources, such as blogs [33] and social networks [34]. SalahEldeen and Nelson [35] found a nearly linear relationship between time and the percentage of lost social media resources.

Research Data and Software

With a focus on research data in particular, independent of the works on Web archiving, there exist various institutional and domain-specific repositories. An overview is given by the Registry of Research Data Repositories[11] [103]. Many of these repositories are operated by universities or have been initiated by research institutes. The University of Edinburgh offers a research data repository, DataShare [104]. It is divided into domain-specific collections which can be searched separately or compositely. Harvard's DataVerse Project[12] supplies a Web service to share, archive and cite research data. RADAR, funded by the German Research Foundation, pursues a service-oriented approach to provide infrastructure and services to host research data repositories [88].

A particular focus of our work has been on software as an example for dynamically evolving and frequently referenced objects. Since software has become an essential part of scientific work, various initiatives for research data management

[10] `https://www.force11.org/about/manifesto`
[11] `http://www.re3data.org/`
[12] `http://dataverse.org/`

have begun to focus on this aspect as well. Peng [105] addresses the need for reproducibility of computational research. Scientific standards for the handling of mathematical software are mentioned by Vogt [106]. On the one hand, researchers must be aware of how to develop software effectively. On the other hand, publishing habits must be adapted to archiving, citing and quality-approved software. An initiative that proclaims this is FAIRDOM[13], where *fair* is an acronym for findable, accessible, interoperable and reproducible [107]. Micro Archives can be considered an attempt to establish these principles, not only for software but for linking and referencing any kinds of dynamic objects through Web archives.

2.2.2 Case Study: Referencing Software on the Web

Software is used in science among all disciplines, from analysis software and supporting tools in the humanities, over controlling and visualization software in medicine to the extensive use of all kinds of software in computer science as well as mathematics. However, referencing software in scientific publications has always been challenging. One reason is that software alone is often not considered a scientific contribution and therefore, properly citable publications do not exist. This is particularly an issue if the software does not tackle a concrete research question but was created as tool for various purposes or different kinds of research, such as standard software like *Microsoft Excel*. Another issue is, even if the software in question is well-known or even published, it undergoes dynamics and the version an article refers to might be different from the one currently available. Furthermore, a name like *Microsoft Excel* refers to a **product** rather than a concrete version of that software. Also, publications typically deal with the innovation and benefits created by the software as a product rather than a concrete build, version or setup. However, this very specific **artifact** may be crucial to reconstruct a software instance as in the original experimental setup, to reproduce experiments and comprehend scientifically published results.

As an example consider the famous bug of *Excel 2007*, which produced the number `100,000` in a cell of which the underlying data equaled to `65,535`[14]. To fix this, the appropriate patch was released shortly after[15], which resulted in an artifact with an updated minor version number, but of course did not update the major version *2007*. Actually, even though 2007 in this case is already more specific than just the product's name, it should be better considered a **sub product** of the **product family** *Excel* rather than a concrete version, since it does not refer to a concrete artifact. Therefore, to verify results of `100,000` in scientific experiments with *Excel 2007* involved, the precise version number referring to the exact artifact used in the experiment is required, but very unlikely to be mentioned in a publication.

In the context of our project *FID Math*, aiming for a mathematical information

[13] `http://fair-dom.org`

[14] `http://blog.wolfram.com/2007/09/25/arithmetic-is-hard-to-get-right` [from 25/09/2007]

[15] `https://support.microsoft.com/en-us/kb/943075` [from 09/10/2007]

service infrastructure, we are facing the problem of referencing software with a focus on all kinds of mathematical applications, tools, as well as services. In the area of math, software is heavily used for various purposes, such as calculations, simulations, visualizations and more. Very often multiple are combined, while the critical task is performed by a script, which is software in itself, running inside an environment like MATLAB, Mathematica or Sage. Settings like these make it particularly challenging to reference a consistent state of the incorporated software. Further, the mix of open source and proprietary software introduces an additional challenge due to crucial differences in many aspects, such as code contributions, licensing, as well as the question for preservation. To address these difficulties in a universal manner, we propose Web archiving as a solution to preserve representations of software on the Web as surrogates for future reference.

Problem and Questions

In an ideal world, the results of every experiment conducted and published by scholars in their scientific work should be reproducible. This in turn implies that every software can be recovered in the exact state as used in their experiments. This either requires a detailed reference to the software's state and general access to software artifacts, or, alternatively, ways to freeze and preserve a software's state and provide it as attachment of a publication. Both seems unrealistic for practical as well as legal reasons. While open source projects often suffer a reliable release process with proper versioning, every committed state is usually precisely identified by a single hash, such as the *SHA* used by GIT[16]. This hash does not only encode the current state of the software but refers to all previous commits comprising relevant metadata records. By contrast, structured metadata of proprietary software is often more explicit, with the author being the company behind the software and each bugfix or patch presumably increases the minor version number of the software. Accordingly, both types of software potentially allow referring to concrete artifacts. The challenge is to establish a unified representation and ways to recover the referenced software.

Open source licenses commonly allow redistribution, which facilitates sharing preservation copies with publications to replay experiments. For proprietary software this is usually considered piracy. Even more difficult to handle are Web APIs and services, where the user does not have access to the actual software, but only to the interface. However, by recovering in this context we do not necessarily mean to obtain a copy of the software, which is only required for replaying experiments and in many cases not usable without proper documentation anyway. Recovering can also mean to get an understanding of the software, for example through its *documentation*, *source code*, *related publications* or *change logs*. Already a brief description can be difficult to obtain though, as referenced software, after many years, might not even exist anymore.

Since the Web can be considered our primary source of all kinds of information today, our hypothesis is that most of the information listed above is available on the

[16] https://git-scm.com

Web as well. Therefore, a snapshot of the corresponding websites of a software from when it was acquired for scholarly use, whether by downloading a copy or ordering in a shop, would constitute a representation of the software at that time. Although it might not include the artifact itself, it is the most comprehensive representation we can get, given the practical and legal restrictions. Hence, it can be considered a temporal surrogate of the actual software.

To realize a Web archiving solution for such a purpose, which enables reliably referencing software surrogates on the Web in scientific publications, we need to overcome a number of challenges. The system has to be aware of all relevant resources of a software and it has to ensure that these resources are preserved at the time of scholarly use. Towards this, we propose Micro Archives in Section 2.2.4 and our Micrawler tool to create such archives in Section 2.2.5. However, with existing Web archives we can already create a solution based on the publication dates of articles using software. In this study we ask the following questions to analyze the applicability of archiving software surrogates on the Web:

(**Q1**) How well is software represented by its surrogate on the Web?
(**Q2**) Which information of software is available on the Web?
(**Q3**) How many websites of mathematical software are archived?
(**Q4**) For how many of these can referenced versions from the past be recovered?

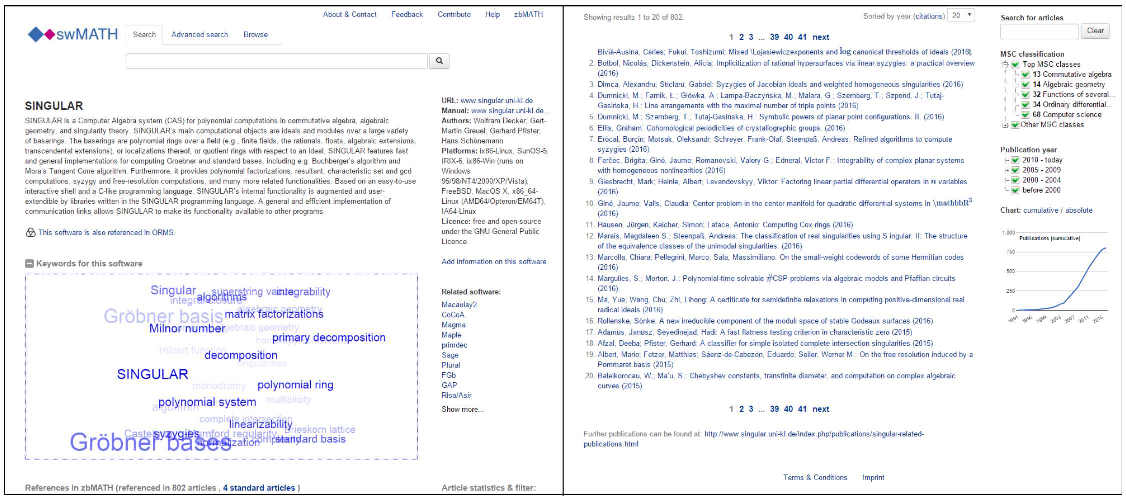
Data and Methodology

The primary source for this work was swMATH, an information service for mathematical software[17] ((M)SW). Based on the information of SW in this directory, we analyzed the linked URLs on the current Web as well as in a Web archive, provided by the Internet Archive.

swMATH in a Nutshell. swMATH is one of the most comprehensive information services for MSW [108]. It contains more than 12,000 records, each representing a SW *product* or *product family* with a unique identifier, as shown in Figure 2.7. swMATH is based on the database of zbMATH[18], one of the most comprehensive collections of mathematical publications, with more 110,000 articles referring to MSW. The biggest challenge for a service like swMATH is to recognize these references. In many cases, only a name is mentioned, while a version or an explicit label as (M)SW is missing. swMATH tackles this with simple heuristics, by scanning titles, abstracts, as well as references of publications to detect typical terms, such as *solver*, *program*, or simply *software*, in combination with a name.

After new candidates have been detected, they are checked manually to ensure the high quality of the service. As part of this manual intervention step, additional metadata, such as the **URL** of a SW is added. Later on, websites are periodically checked and outdated URLs are removed or replaced. In case there is no permanent

[17] http://www.swmath.org
[18] http://www.zbmath.org



SINGULAR

SINGULAR is a Computer Algebra system (CAS) for polynomial computations in commutative algebra, algebraic geometry, and singularity theory. SINGULAR's main computational objects are ideals and modules over a large variety of baserings. The baserings are polynomial rings over a field (e.g., finite fields, the rationals, floats, algebraic extensions, transcendental extensions), or localizations thereof, or quotient rings with respect to an ideal. SINGULAR features fast and general implementations for computing Groebner and standard bases, including e.g. Buchberger's algorithm and Mora's Tangent Cone algorithm. Furthermore, it provides polynomial factorizations, resultant, characteristic set and gcd computations, syzygy and free-resolution computations, and many more related functionalities. Based on an easy-to-use interactive shell and a C-like programming language, SINGULAR's internal functionality is augmented and user-extendible by libraries written in the SINGULAR programming language. A general and efficient implementation of communication links allows SINGULAR to make its functionality available to other programs.

This software is also referenced in ORMS.

Keywords for this software

References in zbMATH (referenced in 802 articles , 4 standard articles)

URL: www.singular.uni-kl.de
Manual: www.singular.uni-kl.de...
Authors: Wolfram Decker; Gert-Martin Greuel; Gerhard Pfister; Hans Schönemann
Platforms: ix86-Linux, SunOS-5, IRIX-6, ix86-Win (runs on Windows 95/98/NT4/2000/XP/Vista), FreeBSD, MacOS X, x86_64-Linux (AMD64/Opteron/EM64T), IA64-Linux
Licence: free and open-source under the GNU General Public Licence.

Add information on this software.

Related software:
Macaulay2
CoCoA
Magma
Maple
primdec
Sage
Plural
FGb
GAP
Risa/Asir
Show more...

Article statistics & filter:

Showing results 1 to 20 of 802.

Sorted by year (citations) 20

1 2 3 ... 39 40 41 next

Bivià-Ausina, Carles; Fukui, Toshizumi: Mixed \Lojasiewiczexponents and log canonical thresholds of ideals (2016)
2. Botbol, Nicolás; Dickenstein, Alicia: Implicitization of rational hypersurfaces via linear syzygies: a practical overview (2016)
3. Dimca, Alexandru; Sticlaru, Gabriel: Syzygies of Jacobian ideals and weighted homogeneous singularities (2016)
4. Dumnicki, M.; Farnik, Ł.; Główka, A.; Lampa-Baczyńska, M.; Malara, G.; Szemberg, T.; Szpond, J.; Tutaj-Gasińska, H.: Line arrangements with the maximal number of triple points (2016)
5. Dumnicki, M.; Szemberg, T.; Tutaj-Gasińska, H.: Symbolic powers of planar point configurations. II. (2016)
6. Ellis, Graham: Cohomological periodicities of crystallographic groups. (2016)
7. Eröcal, Burçin; Motsak, Oleksandr; Schreyer, Frank-Olaf; Steenpaß, Andreas: Refined algorithms to compute syzygies (2016)
8. Ferčec, Brigita; Giné, Jaume; Romanovski, Valery G.; Edneral, Victor F.: Integrability of complex planar systems with homogeneous nonlinearities (2016)
9. Giesbrecht, Mark; Heinle, Albert; Levandovskyy, Viktor: Factoring linear partial differential operators in n variables (2016)
10. Giné, Jaume; Valls, Claudia: Center problem in the center manifold for quadratic differential systems in \mathbbR^3 (2016)
11. Hausen, Jürgen; Keicher, Simon; Laface, Antonio: Computing Cox rings (2016)
12. Marais, Magdaleen S.; Steenpaß, Andreas: The classification of real singularities using S ingular. II: The structure of the equivalence classes of the unimodal singularities. (2016)
13. Marcolla, Chiara; Pellegrini, Marco; Sala, Massimiliano: On the small-weight codewords of some Hermitian codes (2016)
14. Margulies, S.; Morton, J.: Polynomial-time solvable #CSP problems via algebraic models and Pfaffian circuits (2016)
15. Ma, Yue; Wang, Chu; Zhi, Lihong: A certificate for semidefinite relaxations in computing positive-dimensional real radical ideals (2016)
16. Rollenske, Sönke: A new irreducible component of the moduli space of stable Godeaux surfaces (2016)
17. Adamus, Janusz; Seyedinejad, Hadi: A fast flatness testing criterion in characteristic zero (2015)
18. Afzal, Deeba; Pfister, Gerhard: A classifier for simple isolated complete intersection singularities (2015)
19. Albert, Mario; Fetzer, Matthias; Sáenz-de-Cabezón, Eduardo; Seiler, Werner M.: On the free resolution induced by a Pommaret basis (2015)
20. Baleikorocau, W.; Ma'u, S.: Chebyshev constants, transfinite diameter, and computation on complex algebraic curves (2015)

1 2 3 ... 39 40 41 next

Further publications can be found at: http://www.singular.uni-kl.de/index.php/publications/singular-related-publications.html

Search for articles

MSC classification
- ✓ Top MSC classes
  - ✓ **13** Commutative algebra
  - ✓ **14** Algebraic geometry
  - ✓ **32** Functions of several...
  - ✓ **34** Ordinary differential...
  - ✓ **68** Computer science
- ✓ Other MSC classes

Publication year
- ✓ 2010 - today
- ✓ 2005 - 2009
- ✓ 2000 - 2004
- ✓ before 2000

Chart: cumulative / absolute

Figure 2.7. Mathematical Software Singular on swMATH

link that points to a website of the SW, the URLs of a corresponding repository record or a publication is used instead.

Another important feature for our analysis is the **publication list** for every SW on swMATH. Each article in this list is annotated with its publication year. The publications can be sorted chronologically or by the number of citations an article has received. In swMATH, publications also serve as source for additional information, such as related software and the keyword cloud shown in every record (see Fig. 2.7).

In order to enhance the functionality of swMATH, one goal is to capture the dynamics of (M)SW as reflected by the publications over time. We address this aspect by investigating which information of SW is available on the Web and can be recovered from Web archives. Of importance for this study are **URLs** as well as the **publications** of a MSW, which are both available through swMATH.

Analysis. As an initial step for the analysis of each addressed question in our study, we crawled the required datasets using Web2Warc[19], resulting in the following four collections, listed with the last time of crawl:

- **swMATH records** *(28/01/2016 - 14:14:53)*
  All 11,785 software pages available on swMATH at that time.
- **URLs** *(18/02/2016 - 18:55:03)*
  All webpages linked by the 11,125 URLs extracted from the swMATH records.
- **Publications** *(19/02/2016 - 08:30:15)*
  The top 100 user publications with respect to their number of citations received, for all swMATH records. These lists are dynamically loaded into the swMATH records and constitute therefore separate resource to be crawled.
- **Internet Archive** *(22/02/2016 - 19:38:55)*
  Metadata of the captures in the Internet Archive's Web archive for all URLs

[19]https://github.com/helgeho/Web2Warc (*Last commit 73f0934 on Jan 29, 2016*)

Class	Segments
source code	*code, gpl, lgpl, {R-project}, {github}, {googlecode}, {sourceforge}, {cpc}, {gpl}, {bitbucket}, {gnu}*
publications	*publications, papers, journals, publication, article, journal, doi, articles, library, bib, reports, {acm}, {springer}, {sciencedirect}, {wiley}, {cpc}, {arxiv}, {googlebooks}, {ieee}, {doi}, {manuscriptcentral}, {tandfonline}, {oxfordjournals}, {citeseerx}*
updates	*changelog, history, news, blog*
documentation	*doc, documentation, manual, api, reference, handbook, handbuch, referenz, doku, dokumentation, wiki, docs, readme,* ***publications***
artifacts	*exe, zip, gz, tar, download, tgz, files, downloads, ftp,* ***source code***

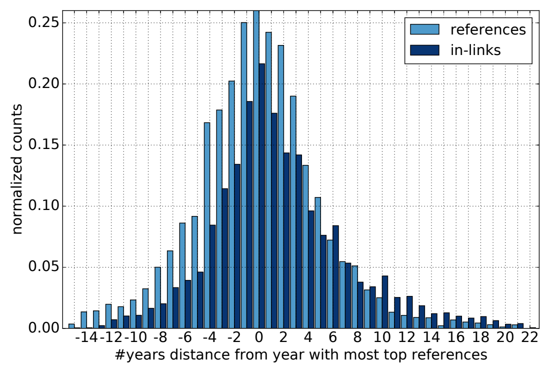
Table 2.8. Segments extracted from software URLs and grouped into classes.

extracted from swMATH, using the *Wayback CDX Server API*[20]. For each URL we fetched the latest capture as well as the one closest to the time of the best publication of the corresponding software with respect to number of citations received.

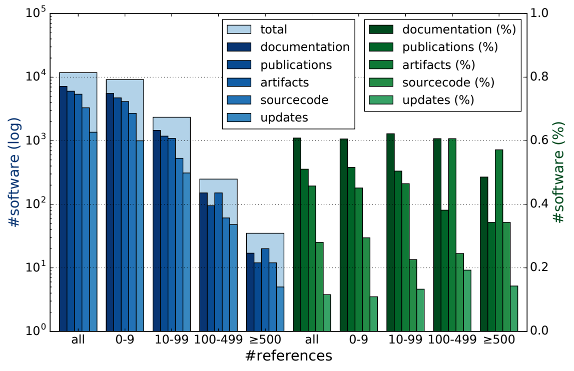
To analyze these collections, which are small Web archives in themselves, we employed ArchiveSpark[21], a framework for Web archive analysis [9]. This way, we extracted the data of interest, such as URLs and publication dates from the swMATH records as well as linked URLs from the crawled webpages. Subsequently, URLs were analyzed further to identify what kind of resources they point to. We split the URLs by means of the following rules: 1. the host is split at dots [.], 2. the path is split at [/.-_], 3. the query string is split at [?=+&:-_]. As indicated by the segments we obtained through those splits, the URLs were classified as different resource types. The classes and segments are shown in Table 2.8. Segments in curly brackets denote whole URLs that match predefined URL patterns, such as GitHub URLs as denoted by *{github}*. These, for instance, are an indicator for available *source code*. Additionally, *documentation* and *artifacts* include all *publications* and *source code* respectively (bold in Table 2.8), since we consider publications to be documentation, and source code implicitly constitutes an artifact of software. Even though these heuristics are by no means complete, we tried to cover all cases that we observed by investigating the URLs manually and are convinced to convey a representative round-up of the available resources with this approach.

In addition to the listed datasets, we collected in-links from external websites to the URLs under consideration. These were extracted from the archived German part of the Web under the top-level domain `.de` from 1996 to 2013, which we had full local access to. Due to scope of this dataset, we performed the in-link analysis only on URLs of domains ending in `.de` as well, assuming that these are better represented in the dataset than arbitrary URLs.

[20] `https://archive.org/help/wayback_api.php`
[21] `https://github.com/helgeho/ArchiveSpark` (*Last commit acc5a16 on Feb 17, 2016*)



(a) Software references vs. in-links.



(b) Information on software webpages.

Figure 2.8. Software Surrogates on the Web

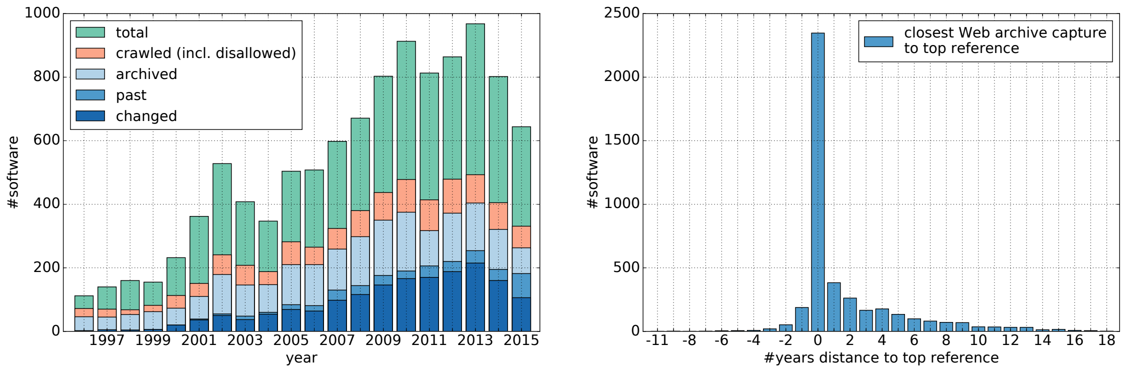
Analysis Results

The results of our study answer the questions introduced before. While **Q1** and **Q2** focus on software (SW) on the Web in general, **Q3** and **Q4** address its coverage by Web archives. In terms of terminology, the website of a SW, addressed in the first two questions, generally represents the *product* or *product family*. At the same time, information on a website usually refer to the current *artifact* of the corresponding SW. The versions available in a Web archive, addressed by the latter questions, are considered surrogates for the artifact that was prevailing at the time of capture.

Software on the Web. The first objective of our analysis was to investigate whether the Web reflects real SW at all and how well webpages as potential surrogates represent actual SW (**Q1**). Our attempt to show such a relation involves the publications referring to a SW over time as well as the in-links to a SW's webpage, which we consider the equivalent to scientific citations on the Web. Figure 2.8a illustrates this remarkable correlation, with references slightly ahead of the in-links. The plot has been normalized by the highest number of publications and in-links for a given SW and aligned by the year with most publications for a SW at $x = 0$. It is based on the links extracted from `.de` pages as well as the number of articles in a year among the top 100 publications. Due to the available link data, only SW with URLs under `.de` was considered.

The fact that the Web lacks slightly behind references in literature suggest that the scientific use of SW leads to visibility and has a strong impact on its popularity. This motivates our effort to archive SW's webpages at the time of publication or even the time of use for a publication. Without such a preservation copy, links that were created as a result of a publication may become stale as the SW and its website evolve and the original reference target cannot be recovered. The same applies to SW references in articles, too.

Next, we asked the question of which information can be obtained from the



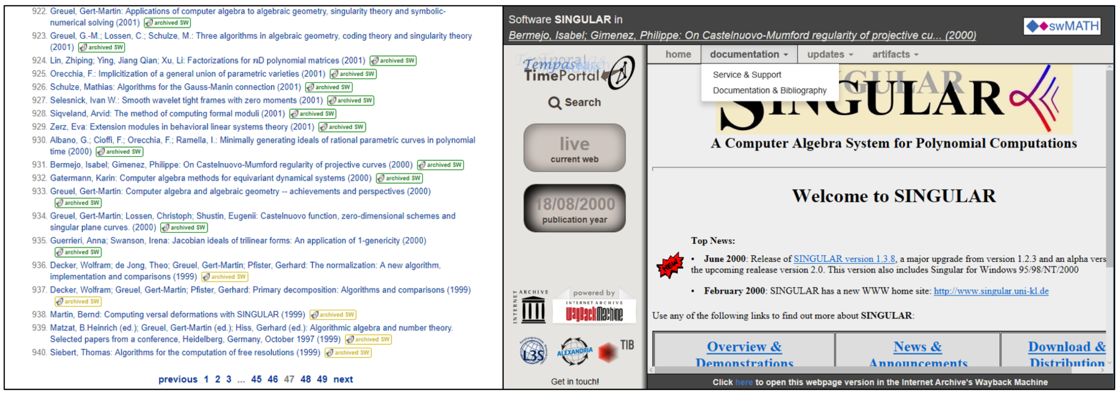
(a) Archived software pages. **(b)** Temporal archive gap.

Figure 2.9. Software Surrogates in Web archives

pages (**Q2**), based on the segments identified in the URLs (see *Data and Methodology* above). Figure 2.8b shows the total numbers as well as results separated by popularity with respect to the number of publications referencing a SW. Interesting here is the finding that for around 60% of the analyzed SW, the corresponding webpage links to some sort of documentation. In many of these cases, publications are available too, or comprise the documentation. However, this changes for more popular SW, where a larger fraction of their documentation is independent of publications. Also, as the plot shows, 50% provide artifacts online, which is again even higher for popular SW. As a result, this trend suggests SW that is well represented on the Web is more prominent and more often used. Therefore, it was surprising to us that we only found source code for around 30% throughout all popularities. Since only this minority enables a detailed tracing of the development process, it is even more important to store temporal copies of the SW's webpages to get a sense of the development through the available information, such as documentation or update reports. Overall, we can conclude that SW webpages contain valuable information and indeed can serve as surrogates of the actual SW.

Software in Web archives. For future work, we are planning to develop strategies and mechanisms to create Web archive collections tailored to SW, which cover the above presented information of SW on the Web. However, for SW published and referenced until then, it is valuable to investigate how well existing, generic Web archives have captured SW surrogates on the Web (**Q3**). The required data for this analysis was obtained from the Internet Archive. Figure 2.9a shows that almost consistent over time the URLs of around 50% of the SW under consideration have been captured in their Web archive. 10% of these were disallowed to be preserved by the *robots.txt* of that website.

While 40% of preserved contents is still a relatively satisfying number, it gets worse when looking at the fraction of archived captures at the time of the top publication referencing the SW, i.e., the article with most citations (see *past* in



922. Greuel, Gert-Martin: Applications of computer algebra to algebraic geometry, singularity theory and symbolic-numerical solving (2001) archived SW
923. Greuel, G.-M.; Lossen, C.; Schulze, M.: Three algorithms in algebraic geometry, coding theory and singularity theory (2001) archived SW
924. Lin, Zhiping; Ying, Jiang Qian; Xu, Li: Factorizations for nD polynomial matrices (2001) archived SW
925. Orecchia, F.: Implicitization of a general union of parametric varieties (2001) archived SW
926. Schulze, Mathias: Algorithms for the Gauss-Manin connection (2001) archived SW
927. Selesnick, Ivan W.: Smooth wavelet tight frames with zero moments (2001) archived SW
928. Siqveland, Arvid: The method of computing formal moduli (2001) archived SW
929. Zerz, Eva: Extension modules in behavioral linear systems theory (2001) archived SW
930. Albano, G.; Cioffi, F.; Orecchia, F.; Ramella, I.: Minimally generating ideals of rational parametric curves in polynomial time (2000) archived SW
931. Bermejo, Isabel; Gimenez, Philippe: On Castelnuovo-Mumford regularity of projective curves (2000) archived SW
932. Gatermann, Karin: Computer algebra methods for equivariant dynamical systems (2000) archived SW
933. Greuel, Gert-Martin: Computer algebra and algebraic geometry -- achievements and perspectives (2000) archived SW
934. Greuel, Gert-Martin; Lossen, Christoph; Shustin, Eugenii: Castelnuovo function, zero-dimensional schemes and singular plane curves. (2000) archived SW
935. Guerrieri, Anna; Swanson, Irena: Jacobian ideals of trilinear forms: An application of 1-genericity (2000) archived SW
936. Decker, Wolfram; de Jong, Theo; Greuel, Gert-Martin; Pfister, Gerhard: The normalization: A new algorithm, implementation and comparisons (1999) archived SW
937. Decker, Wolfram; Greuel, Gert-Martin; Pfister, Gerhard: Primary decomposition: Algorithms and comparisons (1999) archived SW
938. Martin, Bernd: Computing versal deformations with SINGULAR (1999) archived SW
939. Matzat, B.Heinrich (ed.); Greuel, Gert-Martin (ed.); Hiss, Gerhard (ed.): Algorithmic algebra and number theory. Selected papers from a conference, Heidelberg, Germany, October 1997 (1999) archived SW
940. Siebert, Thomas: Algorithms for the computation of free resolutions (1999) archived SW

previous 1 2 3 ... 45 46 47 48 49 next

Software **SINGULAR** in
Bermejo, Isabel; Gimenez, Philippe: On Castelnuovo-Mumford regularity of projective cu... (2000)

A Computer Algebra System for Polynomial Computations

Welcome to SINGULAR

Top News:

- **June 2000**: Release of SINGULAR version 1.3.8, a major upgrade from version 1.2.3 and an alpha vers the upcoming realease version 2.0. This version also includes Singular for Windows 95/98/NT/2000
- **February 2000**: SINGULAR has a new WWW home site: http://www.singular.uni-kl.de

Use any of the following links to find out more about **SINGULAR**:

Overview & Demonstrations | News & Announcements | Download & Distribution

Click here to open this webpage version in the Internet Archive's Wayback Machine

Figure 2.10. Archived website of the software *Singular*, linked from a publication listed on swMATH, corresponding to the publication year.

Fig. 2.9a). Although we considered full years, only around 20% of the SW pages were preserved in that period (**Q4**). However, due to the efforts at swMATH to replace outdated links, this number may actually be higher but not surfaced by our study. At the same time, this would mean that many of the original URLs have been outdated, which in turn suggests a certain development of the corresponding websites. Either way, our findings show the need for a sophisticated Web archiving infrastructure as part of the scientific SW management process, which assigns preserved material on the Web assigned to a specific SW or its reference in a publication, rather than URLs.

Another large portion that is not covered by the number of pages archived in the past are those webpages that were archived shortly before or after the year of the publication. The good news is, this gap is very small as shown in Figure 2.9b. Most pages were captured in the exact year and the remaining were preserved closely around this time with decreasing numbers further away from the year of publication. Thus, by relaxing the time to identify a SW's surrogate in a Web archive to a couple of years around a publication, we can recover even more. Although the representativeness becomes less accurate this way, it can still be helpful to comprehend SW references or reproduce experimental results.

Not exactly surprising, but notable is the fact that almost all pages with archived captures in the past have changed according to the hash/digest of their archived record (see Fig. 2.9a). This motivates to preserve those copies for future reference. As an initial step and as a result of this study, we have integrated temporal links to software mentioned in publications in swMATH. Based on the publication year of the articles listed in swMATH, links are added to the corresponding archived website of a mentioned software along with a status indicator showing whether a snapshot is available in that year or in another year or not at all[22]. As a viewer, we use an adapted version of the Tempas TimePortal, the result viewer of Tempas v2 (cf. Sec. 2.1), as shown in Figure 2.10 [109]. It displays the archived page in the context of the publication it was mentioned in. In addition to

[22]https://blogs.tib.eu/wp/tib/2017/05/19/what-does-the-internet-know-about-the-development-of-software

that, we add a bar with links detected on the captured page, categorized according to the classes in Table 2.8.

2.2.3 On the Coherence of Web Archives

We have investigated two use case scenarios for which Micro Archives would immediately create a major benefit in their scientific use, i.e., blog articles and software. The question we raise is: *How complete and coherent is the archived Web with respect to related resources linked on the corresponding webpages?* [110]. With the concept of Micro Archives and our tool Micrawler, shown in Figure 2.11, we present an approach to improve this coherence by making sure for an object or entity cited today, all related resources are archived today as well, resulting in a coherent collection to represent the subject entity.

Datasets and Methodology

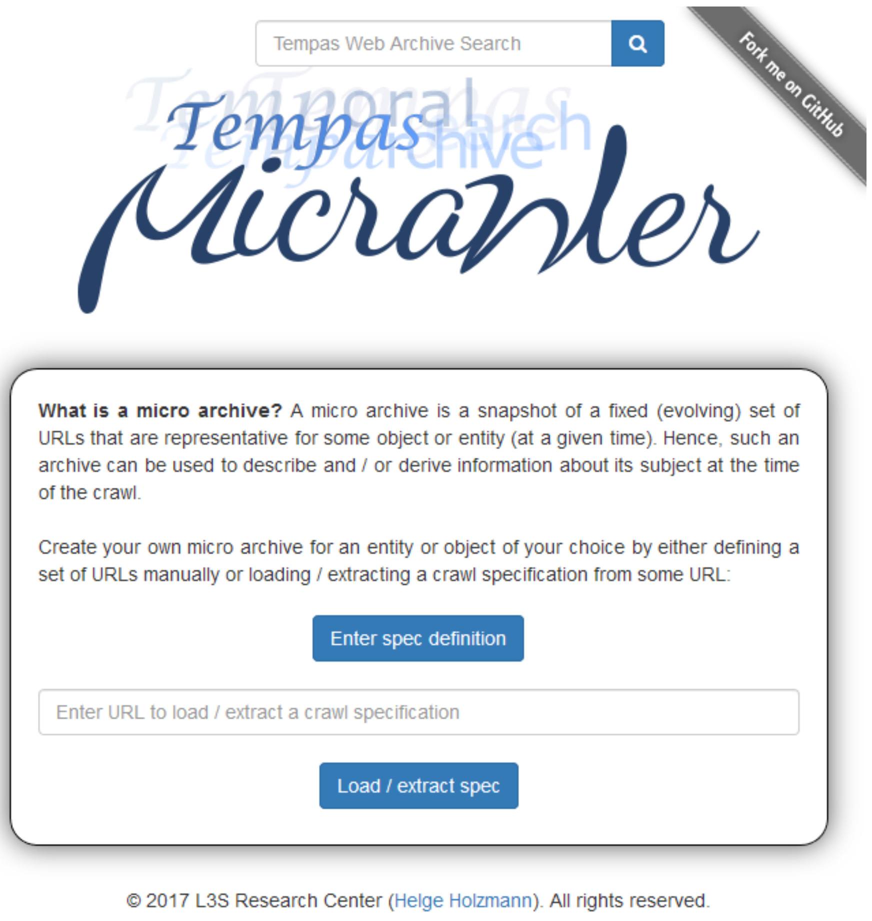
The retrospective analysis of blog articles was done using the *TREC Blogs'08*[23] collection. This corpus consists of 28,488,766 blog posts, collected between 2007 and 2008 for the *TREC 2008 Blog Track*. Hence, we can assume the blog articles to be published during that time period. Although some older ones are included as well, there are definitely no posts composed later than Feb 2009.

As it is more difficult to relate software to a specific point in time, we study its state as of today. For this analysis, we collected all 22,022 URLs[24], each corresponding to a single software, as listed in swMATH's software catalog (see Sec. 2.2.2).

All webpages linked from any of the processed URLs are considered related. Although maybe not complete, we found that many software websites link to corresponding documentation, artifacts, source code and other related artifacts from their homepage [5]. These resources were gathered from the archived snapshot of the corresponding software or blog page. In case of software, we picked the latest captures, and for the retrospective study of blog articles, we picked the earliest snapshot that was available in the Internet Archive's Wayback Machine.

As the process of retrieving an archived snapshot for an URL with all its linked resources is quite time-consuming, we limited our analysis to a random sample of 5,000 objects from each dataset. A single unit of 1 represents a completely archived object with all related resources, the percentage is relative to these. Partially archived objects would be represented by a corresponding floating-point unit. For better readability, the plots have been limited to the 2,134 blog posts and 4,074 software websites for which at least the authority page available, i.e., the actual blog post or representative webpage of a software. Together with the related sources we ended up with a total of 243,336 URLs that we had to fetch for blogs and 123,060 URLs for software, resulting in 48 related resources per blog article and 24 related resources per software on average.

[23] http://ir.dcs.gla.ac.uk/test_collections/blogs08info.html
[24] state at Dec 7, 2017


Tempas Web Archive Search
Fork me on GitHub
Tempas
Micrawler
Enter spec definition
Enter URL to load / extract a crawl specification
Load / extract spec

What is a micro archive? A micro archive is a snapshot of a fixed (evolving) set of URLs that are representative for some object or entity (at a given time). Hence, such an archive can be used to describe and / or derive information about its subject at the time of the crawl.

Create your own micro archive for an entity or object of your choice by either defining a set of URLs manually or loading / extracting a crawl specification from some URL:

© 2017 L3S Research Center (Helge Holzmann). All rights reserved.

Figure 2.11. Micrawler Screenshot

Another fraction was covered by the Web archive but disallowed themselves from being archived through a policy specified in their `robots.txt`. For these, the corresponding objects could not be studied, neither can they be captured with our proposed approach. There are depicted in our plots by the gray bar at the top. For an authority that is archived, but that links to pages that are disallowed, these related resources were ignored.

Each plot contains four lines to show the coverage of the studied objects in the Web archive over time: **resources** represents an object as fraction of its archived resources, **authority** considers the authority pages only, **related** denotes the fraction of *resources* for an object only if the *authority* is archived, and **complete** shows the completely archived ones.

The times shown in the retrospective blog analysis in Figure 2.12 are to be read as when the resources were first archived. From this we can derive the time it takes from publication to the post along with its related resources being archived completely. In contrast, since the analysis of software in Figure 2.13 was conducted from a perspective of today back in time, it should be read from right to left. Hence, they are to be interpreted as the latest available state or version of a software and related resources in the studied Web archive.

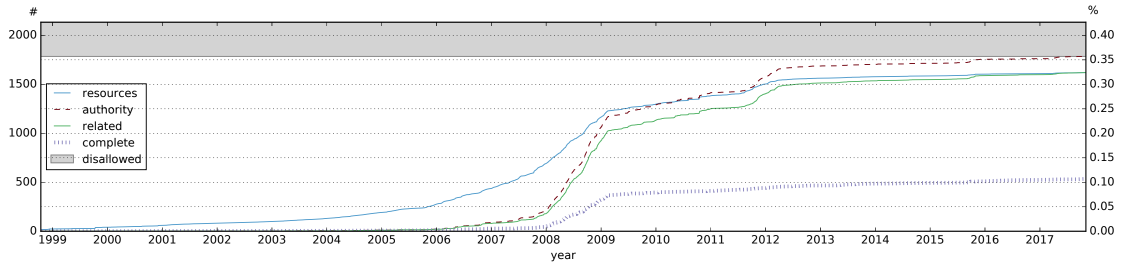


Figure 2.12. Web Archive Timeline: Blogs

Results: Blogs

The timeline in Figure 2.12 shows the results of our study of blog articles. Due to the time of the dataset, which was collected around year 2008, we can observe a major growth in the archive around this time as expected. However, as shown by the resources line, some of the related resources were already preserved long before the blog posts were published, e.g., in 2006 around 5% of the links in an article on average. This makes sense as they have to be online before they are referenced by a blog.

The steep increase of the archived resources to 25% together with the growth of the actual articles (authority pages) indicates that the blogs reference rather recent resources, assuming that they were captured by the archive not too long after publication. This is encouraged by the fact that they were archived slightly before the blog posts, hence, the archive discovered them not through the articles but independently of them.

Once the authority URLs are archived as shown by the dashed line, the related resources go up very closely as well, suggesting that these were indeed already archived before that point. However, although this is a positive finding, it only goes from around 20% at the beginning of 2009 to slightly over 30% today on average for the resources related to the archived authorities, an unfortunately small fraction. The gap to the completely archived articles stays rather large and only reaches about 10% today. This makes us wonder whether actually a coherent and useful impression of the archived blog articles with their hyperlinked references can be obtained from the studied Web archive.

Moreover, after the big increase between 2008 and 2009 still only less than 25% of the blog articles are archived. From there, the lines are growing very slowly, and even today only 35% are preserved with another roughly 10% that were disallowed. Overall, these numbers are not very satisfying.

The use of Micrawler for on-demand archiving of blog articles along with their related resources whenever cited would bring the number of completely archived objects much closer to the number of archived authority pages, providing a more holistic retrospective view onto the blog post as in its referenced state.

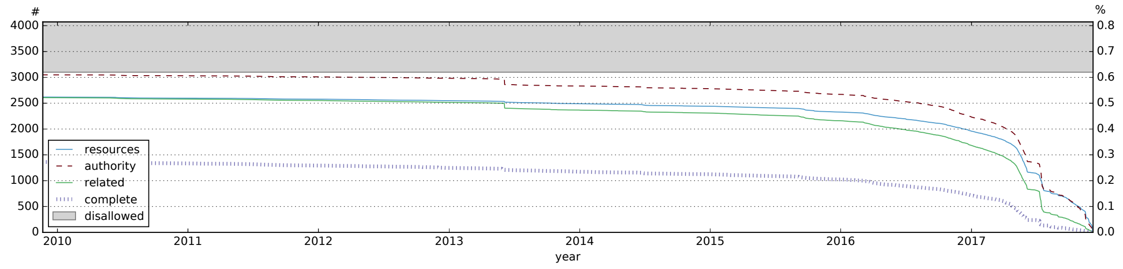


Figure 2.13. Web Archive Timeline: Software

Results: Software

Software on the other hand was studied from its current state, going back until the latest snapshot of a resource had been archived. Immediately noticeable in Figure 2.13 is the larger fraction of disallowed software websites of around 20% as compared to the 10% of disallowed blog articles. However, very positive is the steep growth on the very right of the timeline, resulting in almost 50% of all software authority websites archived already only about one year back from now, at the beginning of 2017. That shows, if an author cites one of this software in his work today, the corresponding website's version found in the Wayback Machine is at most one year old. Unfortunately, there is not much gain by going back in time and even in 2010 and before not more than slightly over 60% are archived overall.

Similar to blogs, the line of complete snapshots is rather low. Hence, from the 50% software sites archived one year ago, for only less than a half are all related resources archived as well. On average, only 40% are preserved at that time as shown by the line of related resources. In the case of software, this is even more severe than for blogs. Blog articles carry a lot of meaning and content themselves, however, authority page of software are usually landing pages that link to the actually meaningful documentation, news, code, etc. The only valuable information on such a software page is often a brief description as well as metadata.

A noticeable difference to the timeline of blogs is that the lines of overall resources and related resources are much closer at any time. In the case of this timeline, which is to be read from the right, that means only a few related resources are recaptured more recently than the corresponding authority page, indicating for the majority of software, the authority page is preserved more frequently than their resources. Other than for blogs, it is quite likely that these are only discovered by the archive crawler through the software websites. That means in practice, the information found on such a landing page are usually more up-to-date than the related resources found by following the links on the archived snapshot.

A snapshot of the software's representation on the Web created using Micrawler would help to improve this situation. It can make sure that for a software cited today all related resources are archived today as well, constituting a rich, complete and coherent digital representation of the software itself.

2.2.4 Micro Archives as Temporal Object Representations

As our case studies have shown, the coherence among related resources in Web archives is not sufficient to reference a consistent state of a represented object. This is what we intent to improve with the introduction of Micro Archives.

There are several applications in which such logical or semantic connections of Web resources that belong to an object or entity would be useful. The most immediate scenario is the use of Micro Archives for referencing rich representations of cited objects in literature or digital works as part of the scientific workflow. Additional use cases that would benefit from Micro Archives are discussed in the end of this Section (see *Opportunities* below).

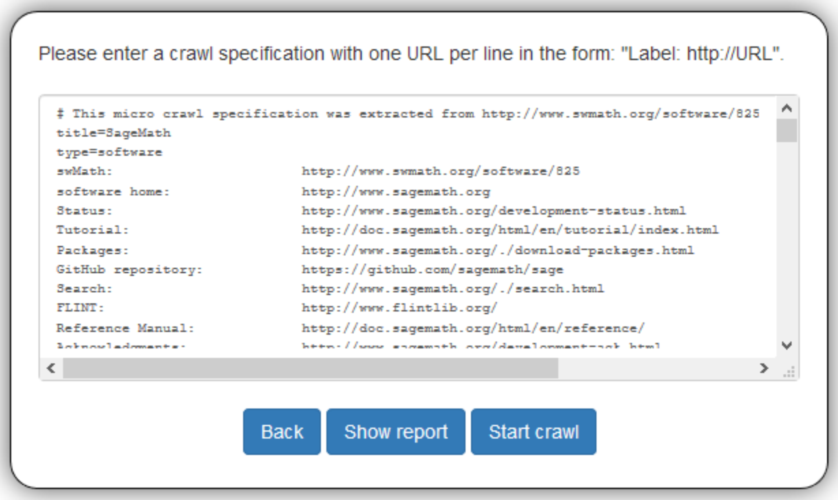
The following steps outline a common workflow to create and cite a Micro Archive. Ideally, this should be done by the authors who cite some object or entity in their work at the time of reference. In case of a blog posts, that is when the article is read, or in case of software, at the time when it is downloaded, so that the used version matches the one represented by the cited Micro Archive.

Specifying Micro Archives

In order to use a Micro Archive as digital representation of any object, it first needs to be defined. Typically, this should not be done by the users manually to avoid an additional overhead to their workflow. However, depending on the concrete application, anyone can specify a Micro Archive with the required set of resources. Resources are identified by their URL along with labels and possibly comments. In addition, a Micro Archive specification should include the name of the represented object as well as additional properties, such as the type, e.g., blog, software, person, company, etc. (see Fig. 2.14). Such crawl specifications can be shared, refined as well as reused. Predefined specifications can be provided or extracted from suitable services, such as repositories or directories, accessible through a dedicated link to cite included items. In case of software, this could be any service that is aware of the relevant URLs, such as a software catalogs like swMATH (see Sec. 2.2.2). A click on this cite link could immediately trigger the archiving process (using a software like Micrawler, see Sec. 2.2.5). To create a Micro Archive of a blog post, the specification can be automatically derived from the links in the post itself.

Crawling / Archiving

Based on the given crawl specification all related resources should be crawled and archived at the same time or with as little delay as possible. Whether only the given URLs are captured or used as seeds for a broader crawl depends on the type of application. In case of blog articles, it might be sufficient to store only the article itself as well as the directly referenced pages. However, in case of software it makes sense to go deeper as the URL of a documentation typically does not contain the entire document but links to chapters and sections, which should also be preserved. The archiving process can be performed by any Web archive, treating each resource


Please enter a crawl specification with one URL per line in the form: "Label: http://URL".

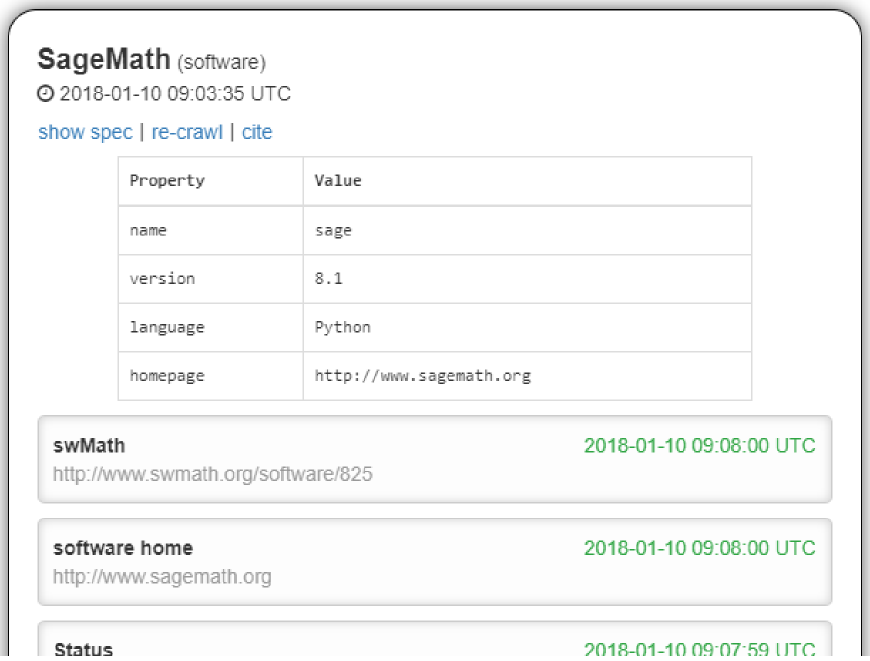
```
# This micro crawl specification was extracted from http://www.swmath.org/software/825
title=SageMath
type=software
swMath:                        http://www.swmath.org/software/825
software home:                 http://www.sagemath.org
Status:                        http://www.sagemath.org/development-status.html
Tutorial:                      http://doc.sagemath.org/html/en/tutorial/index.html
Packages:                      http://www.sagemath.org/./download-packages.html
GitHub repository:             https://github.com/sagemath/sage
Search:                        http://www.sagemath.org/./search.html
FLINT:                         http://www.flintlib.org/
Reference Manual:              http://doc.sagemath.org/html/en/reference/
```

Figure 2.14. Micro Archive Specification Screenshot

as an independent item. Depending on the type of resource, even different archives may be used, like Web archives for webpages, but more software-specific archive for the raw source code. The resulting Micro Archive now serves as an additional layer that connects these captured resources and takes care of a coherent state among them. Hence, an extended version of the original specification, labeled with a timestamp and pointers to the corresponding snapshots in the Web archive, needs to be stored as well, independent of the connected snapshots.

Presentation / Citing

Once created, the Micro Archive is anchored to the time when it was crawled and represents the corresponding object or entity through the resources that were part of the specification. For future reference, a unique handle that is assigned to the Micro Archive, would now be sufficient to cite the preserved state of the represented object. This may be a short URL or more specific identifiers, such as a DOI or others. These would point to a landing page that nicely presents and renders the Micro Archive for the users. Such a landing page lists the state of each capture, whether it was successfully archived or missed, as well as additional metadata from the archive, such as the exact timestamp of the snapshot (see Fig. 2.15). Further, depending on its completeness, the temporal representation of the corresponding object through the included resources is quite rich in information and allows for additional analyses or visualizations. Since the included resources by definition belong and represent a common object, metadata can be extracted fairly easy. If the type of the object or entity is known, this analysis can be even tailored to the specific type. Similarly, more sophisticated queries may be processed dynamically over the contained resources to obtain even more complex information, such as the author of a specific piece of code in a software, which may not be included in a fixed set of meta information that are traditionally used as digital object representations.



SageMath (software)
2018-01-10 09:03:35 UTC
show spec | re-crawl | cite

Property	Value
name	sage
version	8.1
language	Python
homepage	http://www.sagemath.org

swMath 2018-01-10 09:08:00 UTC
http://www.swmath.org/software/825

software home 2018-01-10 09:08:00 UTC
http://www.sagemath.org

Status 2018-01-10 09:07:59 UTC

Figure 2.15. Micro Archive Screenshot

Opportunities

While the primary and most obvious use case scenario for Micro Archives is the temporal representation and reference of objects or entities, we see a lot of potential in such microscopic collections in establishing the missing semantic and logical link among the resources on the Web combined with a temporal embedding:

Supporting Web Archives. An infrastructure around Micrawler that allows for sharing and maintaining crawl specifications as well as existing Micro Archives in combination with a headless implementation that can be triggered programmatically may support Web archives by ensuring coherent snapshots at relevant times. For instance, such a database that is aware of the resources related to an entity would enable publishers or libraries to trigger a snapshot whenever a mention of the entity is detected in a new publication, e.g., all websites and social media accounts of a person can be captured whenever he or she is mentioned in the news. Web archives itself can incorporate this information to prioritize related resources of a page at crawl time as well as use it to improve their access capabilities.

Temporally Relevant Collections. A huge issue in the research field of *Temporal Information Retrieval* [36] and temporal Web archive search [6] is the lack of a ground truth dataset for temporally relevant search results of a query. Micro Archives as a first step towards structuring the Web as well as Web archives in a semantical way constitute exactly such collections for the corresponding entities as queries across time. Hence, a central, curated database as described above, which allows for the retrieval of existing Micro Archives along with the snapshots of related resources would be of importance for these applications and finally enable proper evaluation of temporal retrieval systems. In addition to this, these collections can also be of direct use for the users of Web archives to discover lost webpages from the past.

Structuring the Web. Micro Archives add a semantical as well as a logical structure to Web archives, which represent single entities or objects at different points in time. The identification of such structures along with the existence of archived snapshots for corresponding resources opens up new opportunities in studying the Web. For instance, Web graphs that are typically constructed based on single URLs, hosts or domains, may now be formed according to objects and entities based on their related resources. Scientists would be able to study relations among entities not just based on textual information, which are hard to extract, but based on related resources across time. The coherent snapshots ensure a temporal coverage and realistic topologies in the sub-graphs, which are currently widely broken due to the present incompleteness of Web archives.

Rich Information. A very ambitious and visionary aspect of Micro Archives is the complete reconstruction of represented entities. Wikipedia is a great example of how entities can be represented on the Web. It is not only used for reading and learning about facts, but even to link and disambiguate entity mentions on the Web or in machine learning tasks. However, Wikipedia articles are not written from scratch, they are rather compiled of information found all around the Web, indicated by the many references in these articles. Thus, collections and temporal snapshots of related resources that are representative for an entity may allow for automatic generation of such articles or semantic representations like in knowledge bases. Furthermore, these representations are temporal and thus, can reflect the evolution of corresponding entities.

2.2.5 Micrawler Reference Implementation

Micrawler (*Micro Crawler*) is a reference implementation and proof-of-concept prototype to perform the aforementioned steps of creating and citing Micro Archives. It runs the entire pipeline from specifying over crawling to citing and analyzing Micro Archives. Micrawler is a Web application powered by a set of configurable server modules, which are not part of the project itself, however, we provide reference and demo implementations for each of them together with Micrawler. Figure 2.16 gives an overview of the steps performed by Micrawler and how these connect to the modules as explained in the following. The codebase of Micrawler is open source and published under `https://github.com/helgeho/Micrawler`. The running prototype has been deployed to `http://tempas.l3s.de/Micrawler`.

1. **Spec Proxy:** A crawl specification (spec) can be provided to Micrawler in two ways: 1. in a textual form as list of URLs prefixed with labels and additional properties in a predefined format, 2. as a single URI/URL that either points to a textual spec or to a resource from where a spec can be extracted. In the latter case, Micrawler would send the URL to a *spec proxy* service, which is in charge of deriving the textual spec from the given source (see Fig. 2.14). Our demo implementation currently supports the following identifiers:

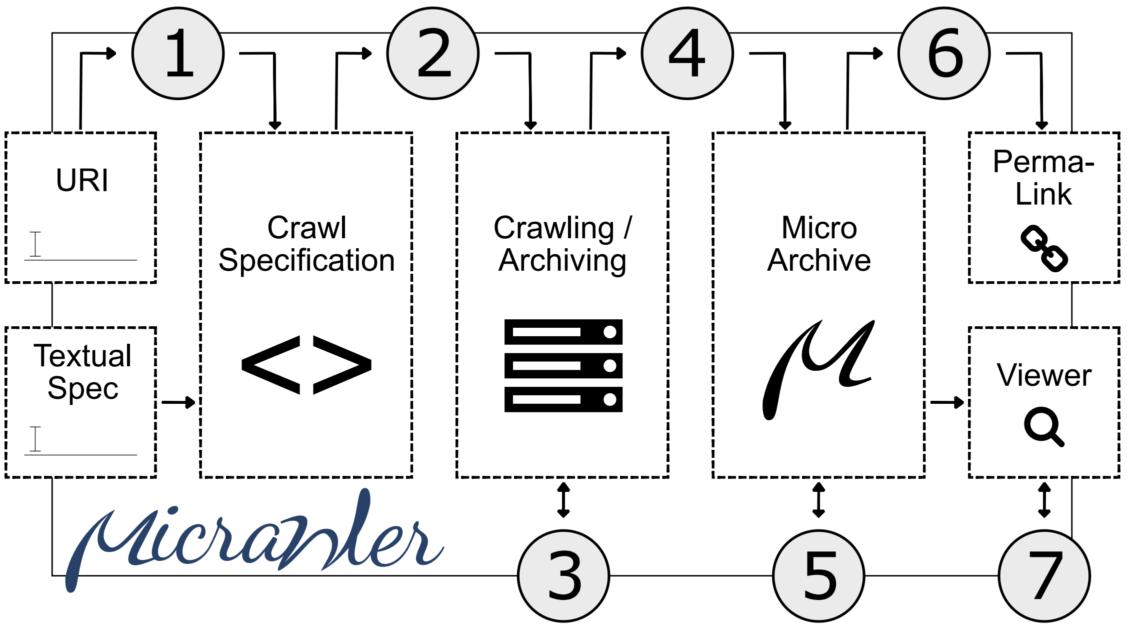


Figure 2.16. Micrawler Architecture and Extension Points / Related Services

- If the given URL refers to a software in swMATH (see Sec. 2.2.2), a corresponding spec is generated from the included software website and linked resources.
- If the given URL refers to a Wikipedia article, a spec is generated for the corresponding entity, containing all external links.
- If the given URL points to an archived resource in the Wayback Machine, an extended spec with the timestamp of the snapshot is created and all linked resources are included. This is treated as a final Micro Archive and hence, the crawling step is skipped.
- As a fallback, e.g., for blog articles, all linked resources from the given URL are extracted and included in the spec. In this case, it neither has a name nor a type, which can be manually added though.

In the future we plan to provide a more sophisticated spec proxy service. For instance, we would like to support widely used blog platforms, like Wordpress, so that the title of an article as well as the type is automatically included in the spec. Further, we are planning a dedicated platform to create, share and maintain Micro Archive specifications for various entities. This way, we would achieve a much cleaner set of URLs for an entity with more descriptive labels than what we currently get from the anchor texts.

2. **Crawl Queue:** Once the Micro Archive is specified so that it well-represents the corresponding object, the spec is sent to another service that generates the queue for the crawl. In the default case, this is exactly the list of URLs as they appear in the specification, which is sufficient for the most common

cases. However, this hook allows for additional customizations and special treatments of certain resources. For instance, if a URL refers to a documentation of some software in a known format, automatically all subsequent URLs of this documentation could be added to the queue and archived as well, while only the homepage of the documentation will be part of the spec to keep it flexible for future reuse. In our demo implementation, the only extension we add is the GitHub metadata API in case the spec contains a GitHub URL. This way it is ensured that all metadata of a software is preserved and can be extracted in the analysis later.

3. **Archiving/Crawl Service:** Each URL in the queue is now sent to an archive. A requirement for this is that the archive provides an on-demand archiving service, such as the *Save Page Now* feature of the Internet Archive's Wayback Machine, which we use in the current implementation. Micrawler allows this to be configured in a very flexible manner. Either every URL will be sent to the same archive or special cases for certain resources can be defined. This way it is possible to tailor the archiving process based on the type of resource. While standard webpages are archived by the Internet Archive, a GitHub repository may be sent to a specialized service, such as *Software Heritage*[25], that does not only preserve the GitHub website, but also copies the raw source code from the repository. In our currently deployed version, all resources are archived at the Internet Archive.

4. **Archive Meta Service:** After all resources in the queue are sent to the respective archives and captured, the Micro Archive is created by enriching the spec with the current timestamp as well as the metadata about each snapshot of the included resource. The *archive meta service* is the one that retrieves these meta information. It double-checks each archived URL against the previously triggered archive and obtains additional information from the index. Hence, this service has to match to the configured archiving services. We currently use the *Wayback `CDX` Server API*[26] (*Crawl Index*) for this, which can be queried for the exact timestamp as well as status information of any resource in their archive.

5. **Analyzers:** The timestamped spec, enriched with meta information from the archive, constitutes the final Micro Archive. The included resources can now dynamically be analyzed and additional information about the represented object may be derived. For this purpose, Micrawler enables the configuration of analyzer services. These can be defined per type to treat Micro Archives for different types of objects differently, as the required mining and analysis processes of may vary among them. For instance, an analyzer for a Micro Archive that represents a software should look for a version number, which is likely to be contained in one of the included resources. While the same method, in case it identifies a version number in a blog post or its related

[25] https://www.softwareheritage.org

[26] https://archive.org/help/wayback_api.php

resources, probably does not constitute metadata of the blog post. If no analyzer is configured for a specific type, this step is skipped.

In our currently deployed demo, Micrawler is only connected to a software analyzer. The analyzer receives the full specification with information about the archived snapshots and returns additional data about the software. In case a GitHub repository is part of the Micro Archive, it checks for a snapshot of the corresponding GitHub metadata API being archived as well and extracts the information from there. If that is not the case, it picks the resources labeled as home and tries to identify version information on its content. Finally, the identified data is returned to Micrawler and displayed in a structured table (see Fig. 2.15). In the future, much more sophisticated visualization methods for more complex information or even dynamic queries will be possible.

6. **Persistence Provider:** In order to share and cite a Micro Archive, it needs to be stored persistently and assigned some handle or identifier to look it up. Although every single resource in the Micro Archive has been captured and stored permanently, the added value is in the semantic or logical connection among the resources. The final timestamped crawl specification with the snapshot information for each included resource is an exact description of this and could be used to share it. However, it is quite long and verbose. Therefore, Micrawler provides an option to cite a Micro Archive. This sends the spec to a persistence provider, which permanently stores the spec and assigns a unique identifier that can be used to refer to the object or entity represented by the Micro Archive. Currently, we store the enriched specs in a text file on our servers and assign it a hash. This is not guaranteed to be permanent though and should not be used in production. In the future we plan to connect a real persistence provider or use the Internet Archive's infrastructure to store the Micro Archive layers as well. Based on the generated handle, Micrawler also provides BibTeX and BibLaTeX records to facilitate the use of Micro Archives in scientific publications as follows [111]:

```
@misc{SageMath,
   title = {{SageMath}},
   type = {software},
   howpublished = {\url{http://tempas.l3s.de/micrawler/permalink/8bcbcec}},
   note ={Archived using Micrawler:  2018-01-10T09:03:35.000Z}
}
```

7. **Viewer:** To view the archived resources of a Micro Archive, they need to be replayed and rendered with all its embeds and assets from the archiving service. Micrawler shows a viewer window for this, in which it loads the resource from the archive, using a configured viewer. Web archives commonly use an instance of the Wayback Machine for this, which can be called by providing the URL and timestamp of the resource to load. Depending on which archive is used or whether different types of archiving services are used for different kinds of resources, the corresponding viewers to be used can be customized.

2.2.6 Conclusion and Outlook

As shown in our case study on software, the Web reflects software to a remarkable extent, with documentation and artifacts on a considerable number of webpages. Hence, the Web can indeed serve as surrogate of actual software. Therefore, as a first step, we have, in collaboration with swMATH, established temporal links to already archived websites for all software on their platform, which is mentioned in scientific articles based on the publication date.

Unfortunately, we found that only for about a half of the analyzed software, the corresponding webpages are currently preserved by an existing Web archive. Further, as we our second study on the coherence of Web archives has shown, only 10% of the studied blog posts and roughly 30% of the analyzed software websites are archived completely, i.e., all linked resources are captured as well. Hence, to fix this in the future, we propose establishing new infrastructures to actively archive webpages as surrogates of software as well as other entities at the time of use or reference. With the concept of Micro Archives, we presented a novel approach to increase these numbers in the future to enable coherent citations. With Micrawler, the author of an article can create such Micro Archives on demand and be provided with a handle to reliably reference these rich object representations. Eventually, explicit temporal references will improve the management of dynamic objects in scientific publications as well as elsewhere on the Web.

Towards this, we want to establish an infrastructure around Micrawler that allows for sharing crawl specifications and existing Micro Archives. We would like to integrate better persistence providers to assign guaranteed permanent identifiers, such as DOIs. Further, a headless implementation of Micrawler would allow for programmatic on-demand archiving to be triggered by third-parties in order to support Web archives and ensure coherent snapshots in the form of Micro Archives.

3 Data-centric View: Analyzing Archival Collections

From a data storage perspective, Web archives commonly consist of two data types: `(W)ARC` and `CDX`. `WARC` is the main format for storing Web archives. It has been specifically designed for the purpose of storing Web resources as well as the requests of a Web archive crawler along with attached headers and meta information. Meanwhile, this has also been standard by ISO[1]. `WARC` is a specialized version of the `ARC` format, which is a general-purpose format for digital archives. Since `ARC` was used in the early days of Web archiving to store Web resources as well, it can be still found in the wild, especially when working with older collections.

`CDX`[2] refers to the *crawl index*, which is not standardized but can be considered a de-facto standard, since it is used by the Wayback Machine and is widely available for most Web archives. `CDX` records are sorted, lightweight representations of the records in a Web archive or the corresponding `(W)ARC` files. They consist only of meta information, such as the type and status of an archived resource and allow for quick look-ups and accesses to the corresponding captures.

These metadata records turn out to be very valuable for both, profiling or analyzing Web archives or the Web of the past at a high, content-agnostic level, as well as for studying parts of the archived contents in depth, by enabling efficient filtering and pre-processing. In the following Section 3.1, we showcase their utility by means of a retrospective study on the dawn of today's popular domains of the archived German Web over 18 years, with a focus on age and size, purely based on `CDX` records. Since `CDX` records are much smaller than the corresponding archives and not subject to legal reservation like the actual contents may be, these files can be easily shared and facilitate the reproduction of similar experiments, which we aim for with the presented framework of formulas and definitions.

With ArchiveSpark, we then present a tool in Section 3.2 to process and analyze archival collections beyond metadata, that makes use of `CDX` records to enable more efficient access to the captures of interest. Its modular architecture can be flexibly

[1] https://www.iso.org/standard/44717.html
[2] http://archive.org/web/researcher/cdx_file_format.php

extended with new functionalities as well as to support different data sources and types, so that it can be even used for other kinds of archives, such as books and journals.

3.1 Retrospective Analysis of Crawl Metadata

The Web is in a state of continuous change, with websites and pages being continuously added, deleted and modified. As previous studies have reported, the Web has been growing and evolving substantially over its lifetime. Researchers have measured and characterized the nature and degree of change in the past [50, 51, 52, 53, 54]. However, these studies primarily focus on content or structural change rates of rather small collections of websites for time periods from a few weeks to a couple of years. One of the interesting findings of such analyses is that a significant part of the changes on the Web are the creation and deletion of pages [53]. With this work we aim to extend those studies with a comprehensive retrospective analysis with a strong topicality, by investigating today's most prominent part of the German Web over an 18-year period from 1996 to 2013. We collected the most popular domains from a diverse set of categories on Amazon's *Alexa* ranking[3] and analyzed on the German Web crawls for this period preserved by the Internet Archive. This makes it the longest study of Web evolution so far.

The dataset gives us the unique opportunity to analyze the evolution of what is popular on the Web today and how those websites have evolved from their early days. At the same time, it puts us into a role similar to an archaeologist, who studies the past only based on what has remained. What remains of the Web in archives is influenced by crawling policies, which are limited due to the available computational resources. Furthermore, not only the Web itself but also the crawlers are subject to evolution. Therefore, we will discuss our assumptions and findings on the Internet Archive dataset in a separate section, which by itself is another interesting contribution of this study and shows the representativeness of the archive with respect to the most popular websites by comparing the growth to the actual Web in terms of registered domains. In this respect, it is interesting to see that those websites are relatively well covered, even though some years back they might not have been as popular as today. This is a positive observation and an important trait of a Web archive since today's popular websites are likely to be looked up by users of an archive from the past as well.

In the following we will use *domain* synonymously for a *website* including its *sub-domains*, e.g., *google.de* and *news.google.de* belong to one website. In contrast, *webpage* is used interchangeably for *URL* and denotes a single *page* of a website.

The questions we ask in our study are inspired by the popular belief about the structure of the Web, but with a focus on the prominent part that people care about most today in Germany:

- *Are popular websites growing old and if so, how can we characterize it?* We

[3] http://www.alexa.com

Evolution and *Domain Age* statistics	
alive$_d(p_i)$	# URLs of d alive in period p_i (were born before t_i and did not die before t_{i+1})
born$_d(p_i)$	# URLs of d born in period p_i (were born after t_i (included) and did not die before t_{i+1})
died$_d(p_i)$	# URLs of d died in period p_i (were born before t_i and died before t_{i+1})
flashed$_d(p_i)$	# URLs of d born and died in period p_i (were born after t_i (included) and died before t_{i+1})
size$_d(p_i)$	Cumulated sizes of URLs of d at the end of period p_i (all URLs that were alive or were born in period p_i)
born_size$_d(p_i)$	Cumulated sizes of URLs of d at the birth of newborn URLs in period p_i
ages$_d(p_i)$	Ages in months of URLs of d at the end of period p_i (all URLs that were alive or were born in period p_i)
URL Age statistics	
count$_d(p_i)$	# URLs of d in period p_i / at age i (were born before t_i and reached age i)
died$_d(p_i)$	# URLs of d that died in period p_i / at age i (were born before t_i and died before t_{i+1})
size$_d(p_i)$	Cumulated sizes of URLs of d at the end of period p_i (only of URLs that did not die in period p_i)
died_size$_d(p_i)$	Cumulated sizes at the death of URLs of d that died in period p_i
died_birth_size$_d(p_i)$	Cumulated sizes at the birth of URLs of d that died in p_i

Table 3.1. Properties Used in the Statistics

were able to confirm what other researchers found earlier: the majority of pages on the Web are rather young. In addition, however, we found that the small long-living fraction contributes significantly to the age, which is increasing.

- *How has the size of popular websites changed over time?* In terms of the volume of a domain, which we define as the number of URLs, we found the growth has been exponential up to now. This is an interesting finding, which we believe is true for the Web in general. Regarding actual sizes, not just existing pages grow, but also newly created ones are larger every year.
- *Do the popular websites from different categories (like business, universities and technology) have different growth rates?* In almost all the conducted analyses we found distinct differences among the considered categories. We find that 75% of the popular university domains of today have been around since *1999* whereas not even 20% of the popular game websites of today were present back then.

Before we present the results of our analysis (Sec. 3.1.3 and 3.1.4), we provide a detailed description of the experimental setup and the measurement metrics used in this study (Sec. 3.1.2). Since all presented properties and statistics are computed purely on metadata from a crawl index (`CDX`), the same analysis can be replicated by other researchers with access to such an index. Using the same definitions (Table 3.1) would allow comparing among datasets, e.g., different national domains. The national top-level domain `.de` constitutes the largest fraction of German-speaking websites, a non-negligible portion of the Web, which we analyze with a focus on the most popular part. The study ends with an analysis and discussion of this dataset as provided by the *Internet Archive* (Sec. 3.1.5).

3.1.1 Related Work

Studying and characterizing change and evolution in the Web falls into the broad field of *Web Dynamics*. Change on the Web can be differentiated into content change and structural change in terms of the Web graph as well as the creation and deletion of webpages. We investigate the latter together with the growth of webpages as a result of content change, which is not analyzed in depth though, as we operated purely on metadata.

By contrast, the earliest studies in this field mainly investigated content changes with respect to change rates. Already in 2000, Cho and Garcia-Molina [50] analyzed 720,000 pages over 4 months in a study motivated by the question on *how to build an effective incremental crawler*. They found that 40% of them change within a week based on their checksum. Similar to us, they focused on popular pages, determined by computing PageRank. In a similar study from 2003, Fetterly et al. [51] analyzed 150 million webpages over a period of 11 weeks with more sophisticated features. They found that 67% of the pages never change, 20% are only minor text changes and 10% of the webpages have changes in the non-textual part. Only around 4% of the webpages report medium to major changes to their text content. The first study in this respect that covers multiple years was done by Koehler [52] in 2002. They analyzed a small sample of 360 pages spanning more than four years from 1996 to 2001 and showed that navigation pages have a better survival rate than content pages. A more fine-grained content analysis was done much later by Adar et al. [54] in 2009, taking hourly and sub-hourly changes into account. They studied page level content changes and tried to capture term-level dynamics on a sample of 55,000 pages with different popularities and different revisitation patterns over 5 weeks. They found that 66% of the visited pages changed during the period under consideration on average every 123 hours.

From a search engine perspective, back in 2004, Ntoulas et al. [53] analyzed the link structure in addition to content of 3-5 million pages over one year. They focused on popular websites once again, according to Google's directory, and observe that 8% of the pages are replaced by newly created ones every week. Out of the remaining about 50% did not change at all during the year under consideration.

With a focus purely on structural change, Baeza-Yates and Poblete [112] investigated the Chilean Web (`.cl`) domain over five years from 2000 to 2004 with questions similar to ours. During this period, their collection grew from 600,000 to 3 million pages. Other studies also focused on national top-level domains, such as `.uk`, which was studied by Bordino et al. [113] in 2008 as well as in a recent study from 2014 by Hale et al. [55]. Bordino et al. [113] analyzed a time-aware Web graph consisting of 100 million pages over one year with monthly granularity. Hale et al. [55] focused on the academic part of the UK under `.ac.uk` from 1996 to 2010 and investigated link patterns. As in our study their collection was also crawled and provided by the Internet Archive. Another recent work by Agata et al. [56] analyzed a collection of 10 million mainly Japanese pages in 2001, which was collected by the Internet Archive as well and is also based on metadata. They report a webpage's average life span of a little more than three years. The most recent study with a national focus was published by Alkwai et al. [57] in 2015. They analyzed around 300,000 Arabic pages in terms of different criteria, such as their coverage on Web archives.

Our work differs from these previous analyses by having a larger temporal coverage as well as new objectives. To this effect, we carry out studies which compare observations across years showcasing evolution of websites in terms of age (see Sec. 3.1.3) and growth both in size and volume (see Sec. 3.1.4).

3.1.2 Setup and Methodology

In our analysis we focused on the aging as well as growth of today's most popular German websites based on a Web archive over 18 years. All information needed for such an analysis are available in the `CDX` metadata index, which most Web archives maintain with their collections.

Dataset Preparations

Our dataset has been provided by the Internet Archive in the context of the ALEXANDRIA project and consists of all their archived text records from the German Web, as defined by the `.de` top-level domain, from 1996 to 2013.

German Web `CDX`. The so-called `CDX` files that we used for our investigation are manifests consisting of all meta information about the crawls in a space-separated format, with one line per capture, i.e. a snapshot of one URL at a given time. The corresponding line in the `CDX` file looks as follows:

```
   <canonical_url  timestamp  original_url
mime_type  status_code  checksum  redirect_url
meta_data  compressed_size  offset  filename>
```

Of importance for this work are the URL, the timestamp, the status code, as well as the size. As the `CDX` files that we used for this analysis only include text files, such as HTML, we could ignore the mime type. Please note that the sizes provided in the `CDX` files corresponding to the records in the archive, compressed in *GZip* format. Therefore, the analysis on sizes does not present the exact sizes of the websites, but trends over time.

In order to handle the large amount of data, we created an index based on the domains as keys. Each domain points to a list of its URLs, where every URL has attached a sub-list with all its captures in the archive in chronological order, including the data as shown above. This allows quick access to all URLs and captures of any available domain.

Today's Popular Domains. There are three types of Web archives. While the first type attempts to preserve a certain part of the Web completely, for instance a national top-level domain, the second type is more focused, aiming for a certain topic or event. Those broad as well as topical crawls are typically done once or periodically without the attempt to capture all changes in between or to preserve the dynamics of the Web. In contrast to that, the third type of Web archives constitutes continuous crawls over a longer time period, which does not claim to preserve everything, but the most important parts according to a certain crawling strategy. This strategy might even change over time to adjust the crawler for a better coverage of a certain aspect. For instance, a typical strategy is to revisit frequently changing pages more often. Therefore, the temporal coverage of some websites in the archive may be very good, while others are missed completely. This selective crawling introduces a certain bias to the archive, which however is difficult to track retrospectively.

Our collection is of the third type, plus, it includes data donations, which were crawled by third-party organizations. For that reason, it does not cover the entire Web, but constitutes a sample biased by the different crawling strategies. Accordingly, a random sample of the collection would again be biased and it will require further research to analyze what the collection actually consists of to create a more representative sample of the entire German Web.

Therefore, instead of sampling we decided to focus on a well-defined subset, which in addition is inherently substantial for users as well as Web crawlers: the today's popular domains from their early stages in 1996 up to now (2013 to be exact). These websites are of interest for most readers and at the same time have the biggest impact on upcoming research on Web archiving, crawling, IR and related areas, since those disciplines typically focus on rather prominent websites. Also, as we will show later (see Sec. 3.1.5), this subset nicely represents the actual growth of the Web in terms of registered domains.

The selection of domains was taken from *Alexa* by fetching the top websites of different categories, like *Business*, *Society*, *Sports* and others. To match our dataset, we only picked those categories listed under German [4]. In addition to the top categories, we also took two sub-categories for news and universities, which we considered especially relevant. As our dataset only consists of domains ending with the German top-level domain `.de` and not all German websites listed on Alexa are under `.de`, we filtered out those websites with another top-level domain. Out of the remaining, we picked the top 100 from every category (or less for smaller categories, like news) to form our dataset. The last time we retrieved the rankings from Alexa was on July, 10th 2014 at 09:26 GMT+1.

[4] `http://alexa.com/topsites/category/Top/World/Deutsch`

Category	**# Domains**	**# Sub-Domains**	**# URLs**
Computer	100	561	2138786
Recreation	100	380	981638
Society	100	368	832017
Health	100	274	453282
Kids & Teens	100	234	311705
Culture	100	250	934552
Media	100	512	1981877
Shopping	100	429	6726195
Regional	100	793	3069791
Games	99	304	718348
Sports	100	290	656859
Business	100	546	1534639
Education	100	827	1240196
Science	100	398	579821
Home	100	325	1762361
News	40	117	820163
Universities	100	828	659175
TOTAL	***1444***	***5846***	***20778475***

Table 3.2. Dataset Details

Dataset extraction. Based on the selected domains from Alexa, we filtered our `CDX` dataset by taking only those records with URLs belonging to one of the domains. Additionally, we cleaned the dataset by discarding the following URLs:

- All URLs ending with one of the following extensions: `.jpg, .png, .gif, .css, .js`, because these constitute embeds and not self-contained resources, like websites. Although the dataset only consists of URLs with mime type *text*, it included image types either because the server returned a wrong type or the files were not available and pointed to an error page.
- All URLs that have never returned a successful HTTP status code (starting with 2). Those are most likely broken links, which the crawler followed, but which did not lead to a successful response.
- All URLs that were not crawled anymore in 2013, i.e., the last year of the dataset, even if the last available capture was successful. Keeping them would result in an inconsistent state, because we cannot tell what happened to them after the last time they were crawled.
- All URLs that have been crawled successfully only once, even if this was in 2013. As it exists only a single capture of those pages, they do not contribute to our evolution analysis at this point. Most likely, the Internet Archive crawler has just begun to crawl them.

Ultimately, we ended up with a dataset consisting of 17 categories with today's popular domains from the German Web, as presented in Table 3.2. The dataset covers in total 1,444 domains with 5,846 sub-domains and more than 20 million URLs (20,778,475 URLs to be exact).

Statistics and Metrics

Our statistics were gathered in two steps. First, a precomputation step counted different properties of a domain. Afterwards, we aggregated these properties into meaningful metrics. The following subsections describe these two steps in detail and define the terminology used in the analysis results (Sec. 3.1.3 and 3.1.4).

We use the terms of *birth*, *death* and *life* to describe the lifetime of a URL or domain in our dataset. We consider a URL or domain to be alive from the time it first appeared in the Web archive until it was last seen online.

Precomputations. For each domain, we precomputed three types of statistics: *Evolution*, *Domain Age* and *URL Age* statistics. Each of them describes a collection of properties, such as *size* and *age*, computed in different units, i.e., calendar years, domain years, URL years. For all statistics, one unit i spans a period p_i of one year time from t_i to t_{i+1} (excluded), which may or may not be a calendar year from 1 Jan to 31 Dec, depending on the type of statistics presented.

We decided not to collect more fine-grained statistics, such as monthly or weekly, because a higher resolution would not have had any advantages for our analysis and is not sufficiently supported by the dataset. While studies on change rates would require more steady crawls, this is not required for an evolution study such as the one we present as the overall trends are not affected. Also, we cannot guarantee such fine-grained captures with our dataset (see Sec. 3.1.5).

The following definitions describe the statistics:

- ***Evolution* statistics**:
  Values are measured per calendar year.
  t_i denotes the beginning of the calendar year i.
- ***Domain Age* statistics**:
  Values are measured for full years starting from the first date a domain occurs in the dataset (e.g., for a domain that appears first in t_0 = 04.05.2000 10:30:45, age $i = 0$ spans from to 04.05.2001 10:30:44).
  t_i denotes the beginning of the domain age i.
- ***URL Age* statistics**:
  Values are measured for full years of the analyzed URLs. As before the statistics are gathered per domain, however, here by combining values of different URLs at the same age.
  t_i denotes the beginning of the URL age i.

Age statistics (*Domain Age* and *URL Age*) do not necessarily reflect the actual age of domains/URLs, but their age as evident from the dataset. These ages probably do not diverge much, but some time might have passed after the creation of a new domain until it is included in the Web archive.

Evolution and *Domain Age* statistics are similar in the sense that both describe the evolution of a domain over time. The *URL Age* statistics on the other hand are relative to the time of a domain's URLs, which reflects different periods of a domain but aggregates URLs at the same age. This enables different kinds of statistics as shown below.

Aggregation. The precomputed statistics were accumulated among the domains in each category as well as among all categories. For the sake of clarity, we present only selected categories in our plots, which best represent the overall observations as well as some outliers. Each metric that we analyze below is defined per period p_i on the set of domains that appeared in this period D_i. For instance, a domain which was born in the year 2000 is not included in D_i for any $i < 2000$ in the *Evolution* statistics. The same applies to *Domain Age* and *URL Age* statistics with i referring to relative years instead of calendar years.

The aggregations with corresponding formulas that we present and discuss in the following are presented along with the plots. The definitions of the used properties are listed in Table 3.1. In addition to the given definition of alive URLs, we define the number of URLs alive at a single time point, which is a special case for a period with length 0: while $\text{alive}_d(p_i)$ is defined for a period $p_i = [t_i, t_{i+1})$ and denotes the URLs that were alive the entire interval, $\text{alive}_d(i)$ refers to the very end of this period. It includes the URLs that were alive during the entire period p_i plus the ones that were born in period p_i:

$$\text{alive}_d(i) = \text{alive}_d(p_i) + \text{born}_d(p_i)$$

3.1.3 The Age of the Web

The Web started around 25 years ago and has been maturing ever since. However, is its actual age really increasing or is its content constantly being refreshed, by pages being added and removed? To answer this question, we analyzed the age of the Web in terms of how long URLs have been existent. It turns out, while the majority of popular webpages are young, older pages are aging further. We show the distribution of ages among URLs as well as the evolution of the long-living parts of the Web.

Distribution

It has been shown by other researchers that most URLs on the Web are rather short living [53, 51], i.e., less than a year. However, nothing could be deduced about the URLs which survived after a year. Also, there was no evidence whether the fraction of these short-term URLs increased or decreased over time. To answer these questions, we first investigate the age distribution to determine what fraction of URLs is short or longer living and how this differs among the different categories. In this analysis, we only consider URLs that died during the timespan of our dataset, determined by an unsuccessful status code without another successful status code thereafter. The end time of such a URL is set to the time of the first returned unsuccessful status code. The begin time of the URL is the time it was crawled first.

Figure 3.1 shows the fraction of URLs per domain that died at age i, averaged over all domains D_i that reached this age. It is defined on the *URL Age* statistics (see *Statistics and Metrics* in Sec. 3.1.2), with p_i referring to the period of a URL's age i:

$$\frac{1}{|D_i|}\sum_{d\in D_i}\frac{\text{died}_d(p_i)}{\text{count}_d(p_i)}$$

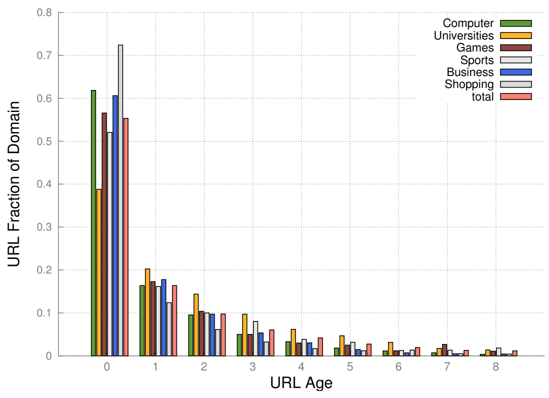


Figure 3.1. URL Age Distribution

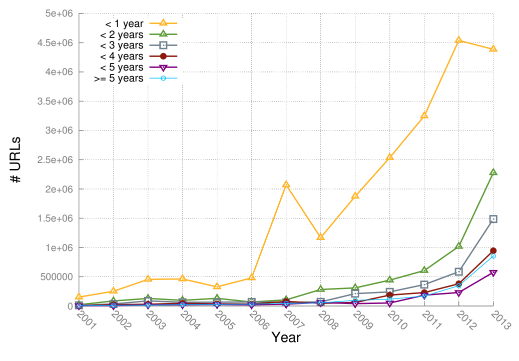
The age distribution shows that, indeed, the largest fraction of the URLs of a domain, about 55%, live less than a year. A considerable fraction of URLs die at the age of two to five. These are what we denote as short-living pages. Every page that lives longer than five years is considered long-living and subject to contribute to the aging of the Web. These constitute the long tail in this distribution. We do not show the entire tail in this figure, but we considered URLs up to ages of thirteen. It is interesting to observe that the university websites have a significantly higher number of URLs dying after the first year, while less than 40% of webpages die at the age of 0. For each of the subsequent ages they consistently outnumber other categories indicating that university webpages tend to be rather long-living. In contrast, we have shopping websites, which have the highest number of pages, 73% of all its URLs, that die within their first year.

Now we turn to the second question of how the overall age distribution evolves over time, presented in Figures 3.2a and 3.2b. For this, we resort to a different style of analysis by considering the number of URLs at a certain age in the given year, instead of how long they lived in the end. We divided the ages into six age buckets of URLs that lived for less than – a year, 2 years, 3 years, 4 years, 5 years and 6 years or longer, which includes the URLs at age five together with the long-living ones. We observe in Figure 3.2a, that over the years the number of URLs for each bucket increases super-linearly. Interestingly, this trend correlates with the domain volume which is presented in the next section.

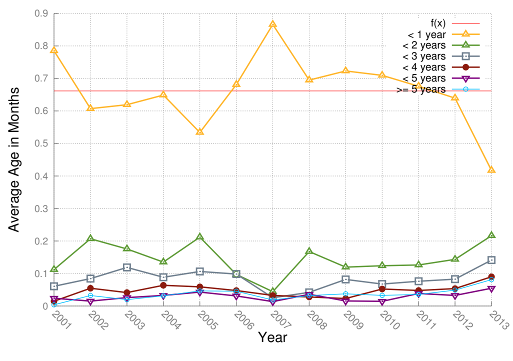
Further, we investigate the normalized distribution for all years in Figure 3.2b. The normalized value of an age bucket α at a given year p_i is defined as follows (on *Evolution* statistics):

$$\frac{\sum_{d\in D_i}|\{a\in \text{ages}_d(p_i)|\alpha\cdot 12\le a<(\alpha+1)\cdot 12\}|}{\sum_{d\in D_i}\text{alive}_d(i)}$$

Although the number of URLs overall grows over the years, as suggested by Figure 3.2a, the fraction of the URLs at different ages remains more or less stable. As emphasized by the computed fitted line in Figure 3.2b, almost 70% of all webpages are younger than a year at any time during the Web's lifetime. The fact that



(a) Total



(b) Normalized

Figure 3.2. Evolving URL Age Distribution

the sizes of all age buckets are equally stable over time suggests that, although the Web is growing, it consists of equal proportions of different aged webpages at any time.

As a result of the retrospective nature of this study, abnormal artifacts that appear in some of the plots are difficult to track. Similar to the peak in year 2007 in Figure 3.2a there are artifacts in the following figures as well. These kinds of abnormalities are most likely due to the different data sources that donated crawls of very diverse volume and size to the Internet Archive. However, as all of them are local phenomena, they do not affect our analysis as the global trend can be clearly recognized in all figures. More details on the dataset are discussed in Section 3.1.5.

Aging

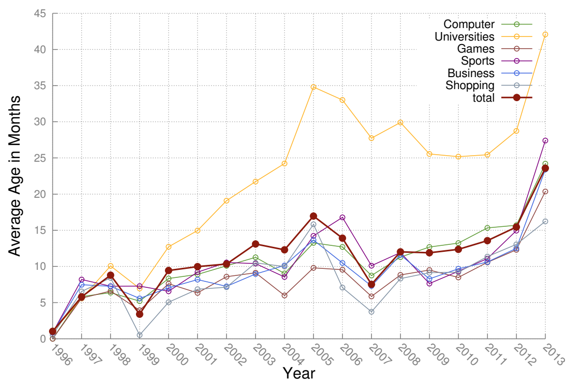
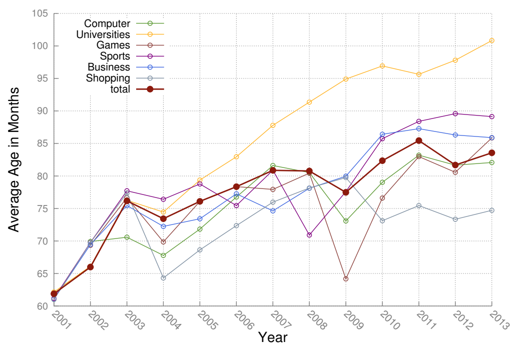


Figure 3.3. URL Age Evolution

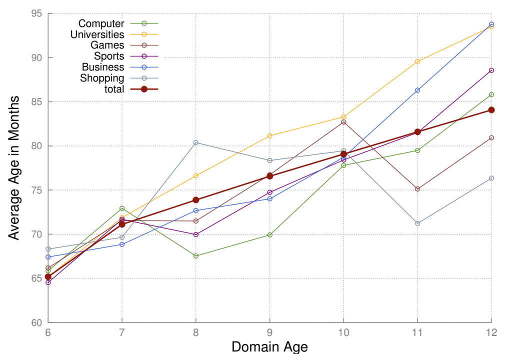
Knowing that the majority of pages on the Web are rather fresh, we now analyze the evolution of the Web's average age. Rather ironically, like humans can grow old but stay younger by eating healthy and doing sports, a similar trend applies to the Web as most of its constituent webpages are frequently replaced. To investigate this, we computed the *average age* of the Web in months at any given year as defined below (on *Evolution* statistics) and plotted in Figure 3.3:

$$\frac{\sum_{d \in D_i} \sum_{a \in \text{ages}_d(p_i)} a}{\sum_{d \in D_i} |\text{ages}_d(p_i)|}$$

The figure shows that the Web is actually growing older after all. While the average age of the Web was about 10 months during the year 2000, it grew almost 50% by the year 2012. This can possibly be attributed to the stability of age



(a) Evolution



(b) Domain Life

Figure 3.4. Age of Long-Living URLs (older than five years)

distributions as shown before. Specifically, the fraction of long-living webpages, which are constantly aging, contributes to a higher age every year.

This aging is almost linear, following the curve $f(x) = a \cdot x + b$, where x is the number of years calculated from 1996. The estimated values for the parameters of this curve are $a = 0.74, b = 4.89$ with an asymptotic error of 8.41% (the corresponding plot is attached in Figure 3.10a). This aging would lead to an average URL age of 23 months in the year 2020, which is double the age of 2005, while the age today or at the end of our dataset (2013) to be exact is 1.5 years. According to this finding, the Web will on average turn three in 2038. However, as our dataset goes back only until 1996, there might be even older pages on the Web. For this reason, our result can be considered as a lower bound.

We further verify our claim that this aging is caused by the long-living pages by analyzing the age of webpages older than five years using the following expression (defined on *Evolution* statistics):

$$\frac{\sum_{d \in D_i} \sum_{a \in \{a \in \text{ages}_d(p_i) | a > 5 \cdot 12\}} a}{\sum_{d \in D_i} |\{a \in \text{ages}_d(p_i) | a > 5 \cdot 12\}|}$$

The corresponding plot in Figure 3.4a visualizes the quite significant growth in age of the long-living URLs. Even though this old part is just a small fraction of the entire Web, its increasing age leads to the slow increase of the Web's actual age that we have shown above. This figure only starts in 2001 as there exist no long-living URLs in our dataset before.

The same observation can be made by analyzing the average age of long-living URLs at a given age of the corresponding domains in Figure 3.4b. This is defined by the same formula as used before, but on *Domain Age* statistics with p_i referring to the of age i of a domain (see *Statistics and Metrics* in Sec. 3.1.2). The plot reflects the actual aging of the popular domains in our dataset in contrast to their real age, as shown on the x-axis: when a domain turns 10 years, their URLs are on average only 80 months old, which is about 6.5 years.

Corresponding to what we observed before, all plots in this subsection acknowledge the characteristics in terms of age for different categories. While websites of universities appear to be the oldest, others such as sports, business and computer websites tend to be much fresher, not to say more up to date.

3.1.4 The Growth of the Web

We now turn our attention to measuring the *size of the popular Web* and how it has evolved over time. The size of the Web can be interpreted as the number of webpages or as the actual size of its content. We refer to the number of websites and pages as the volume of the Web or a domain, while size refers to the actual file size (including markup as well as the content of a page). In this section we study both interpretations and their evolution over time.

By design, we expect growth as we focus on today's popular domains, which have grown popular over time and therefore, have naturally grown in volume and probably size, too. The question now is how this growth, which made the websites as popular as they are today, can be characterized.

Volume

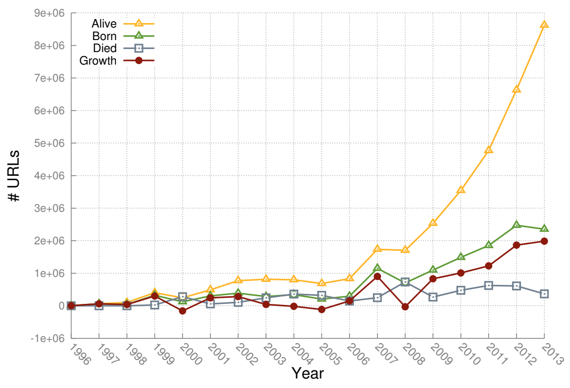
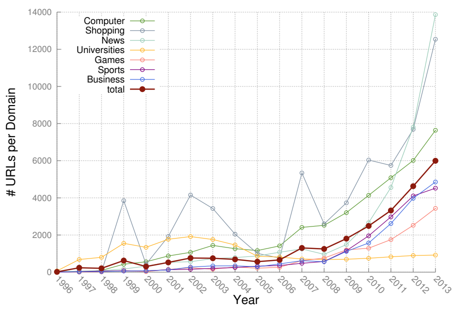


Figure 3.5. Evolution of the Web's URL Volume

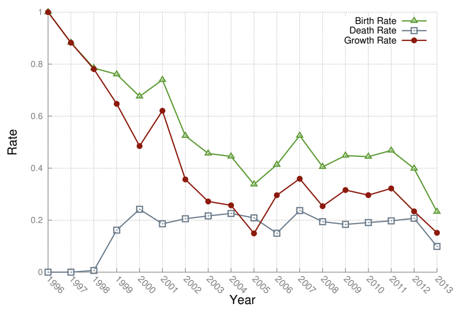
Considering that the number of domains in our dataset grows every year as we will see in Section 3.1.5, it is not surprising that the number of URLs grows as well. However, if this was the only reason, the growth would be similar to the growth of our dataset, which is not the case. We analyzed this by computing four properties: (a) the number of newborn URLs in a year, (b) the number of URLs that died in a year, (c) the number of URLs that are alive at the end of a year, as well as (d) the growth rate. The growth rate is the difference between the number of born and died URLs. While all other numbers are computed over the period of a year p_i, the number of URLs alive is considered at the end of the year i, defined as follows (on *Evolution* statistics):

$$\sum_{d \in D_i} \text{alive}_d(i)$$

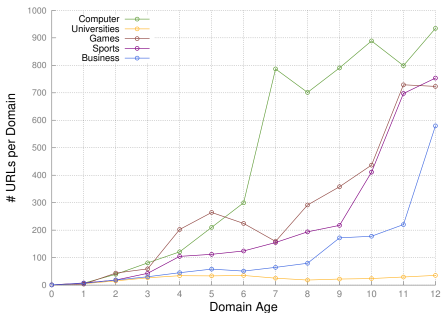
The results are presented in Figure 3.5, which shows that the Web is growing a little faster every year. Especially noticeable is the strong growth starting from 2006, which however might be due to the characteristics of the dataset after all. The reason for this growth of the Web is that there are more new URLs born every year, while the number of dying URLs remains almost constant. In order to affirm that this finding is independent from the growing number of domains in our dataset, we investigated the average number of URLs per domain over the



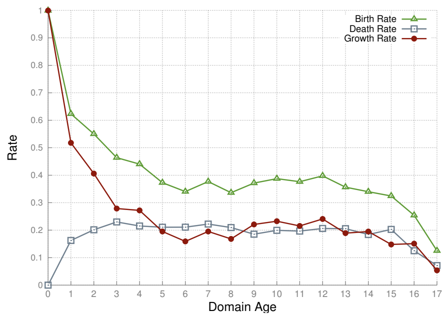
(a) Domain Volume Evolution



(b) Birth/Death/Growth Rates Evolution



(c) Domain Volume over Domain Life



(d) Birth/Death/Growth Rates over Domain Life

Figure 3.6. Domain Volume

years as well. The formula below (defined on *Evolution* statistics) describes this progression, which is shown by the plots in Figure 3.6a per category:

$$\frac{1}{|D_i|}\sum_{d\in D_i}\text{alive}_d(i)$$

Figure 3.6b shows the corresponding average growth rate per domain, as defined below (on *Evolution* statistics), together with birth and death rates. The growth rate describes the difference of born and died URLs of one domain in a given year as fraction of the ones that were alive at the beginning of the year:

$$\frac{1}{|D_i|}\sum_{d\in D_i}\frac{\text{born}_d(p_i)-\text{died}_d(p_i)}{\text{alive}_d(p_i)+\text{died}_d(p_i)}$$

Except for the beginning of this plot, which is most likely due to the transient state at the early years of our dataset, the growth rate is relatively stable at around 30%. Based on this, we can deduce that the number of URLs that are born or die depends on the volume of the Web or their domain. However, among categories the growth varies strongly. While most of them follow the overall trend, university websites barely grow in volume at all, as presented earlier in Figure 3.6a. Even in 2013 they still only consist of about 1,000 URLs on average, whereas computer websites comprise almost 8,000 and shopping as well as news websites more than 12,000 URLs.

The average domain volume follows an exponential curve $f(x) = a \cdot b^x + c$, where x is the number of years calculated from 1996. The estimated values for the parameters of this curve are $a = 22.82, b = 1.38, c = 300.18$ with an asymptotic error of 2.07% (the corresponding plot is attached in Figure 3.10b). Assuming the growth continues with the same rate, in the year 2020 the number of URLs of the popular domains would be almost 6.7 times the number of URLs today (2014) and by 2030 it would be 166 times that of today. Already within the next two years the domain volume would be doubled. Even though this prediction might be weakened due to our crawling assumptions for archives (see Sec. 3.1.5) or the resource limiting is not exponential with the same degree (which is indeed the case as confirmed by the Internet Archive), the exponential nature is still retained, although not as strong.

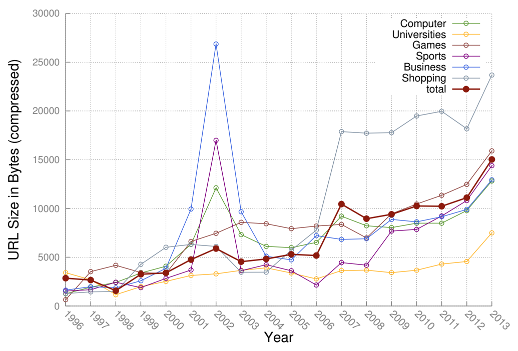
Another perspective to look at the growth of websites is from the age of a domain in contrast to absolute years. Instead of plotting total numbers, this time we analyzed the number of URLs at every age of a domain in relation to its initial volume (defined on *Domain Age* statistics):

$$\frac{1}{|D_i|} \sum_{d \in D_i} \frac{\text{alive}_d(i)}{\text{alive}_d(0)}$$

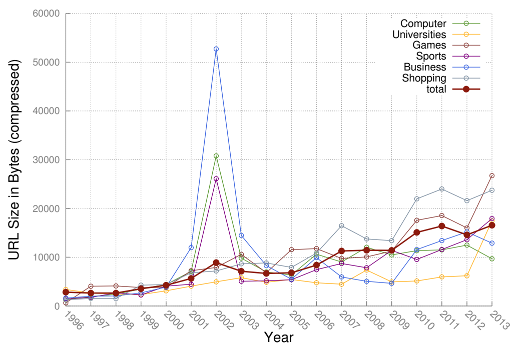
Figure 3.6c gives an impression of this relative volume over the lifetime of a domain for five selected categories. We decided to look only at the first 12 years, as our data is not representative enough for older domains. Most noticeable is a quick growth at some point for the websites in most categories. However, the time of this critical take off varies. While computer websites appear to have a strong growth already very early around year six, where they reach 800 times the volume that they started with at birth, and stagnate afterwards, most categories take longer. As observed before, university websites hardly grow in volume at all. Interestingly, the average growth during the lifetime of a domain, as presented in Figure 3.6d, looks very similar to the actual growth of the popular German Web over time.

Size

Apart from the volume also the actual size in bytes has been growing. We found this to be the result of two evolutions: newborn URLs appear to be larger nowadays than they used to be earlier and, in addition, URLs grow in size during their lifetime.



(a) Alive Size



(b) Birth Size

Figure 3.7. URL Size Evolution

We first analyzed the average size of a URL evolving over time (defined on *Evolution* statistics):

$$\frac{\sum_{d\in D_i} \text{size}_d(p_i)}{\sum_{d\in D_i} \text{alive}_d(i)}$$

Figure 3.7a shows that the size of URLs indeed has increased over the years. This can either mean that websites today consist of more content than they used to in earlier days of the Web, or the markup has grown.

As it turns out, a major growth in size is contributed by newborn URLs, as defined below (on *Evolution* statistics):

$$\frac{\sum_{d\in D_i} \text{born\_size}_d(p_i)}{\sum_{d\in D_i} \text{born}_d(p_i) + \text{flashed}_d(p_i)}$$

This evolution, presented by Figure 3.7b, is similar to the overall growth in size. Its trend follows a linear curve $f(x) = a \cdot x + b$, where x is the number of years calculated from 1996. The estimated values for the parameters of this curve are $a = 866, b = 1320$ with an asymptotic error of 6.9% (the corresponding plot is attached in Figure 3.10c). Based on this, in the year 2038 a new URL will be born on average with double the size as today (2016). As these are compressed sizes (see Sec. 3.1.2), we cannot state actual numbers though.

Another factor that contributes to the growth of URL sizes is the growth of existing URLs during their lifetime. For this analysis we only took those URLs into account that died at some point within the period of our dataset and computed the average size at birth and at death of all URLs that reached a certain age, as defined by the formulas below (on *URL Age* statistics):

$$\frac{\sum_{d\in D_i, j\geq i} \text{died\_birth\_size}_d(p_j)}{\sum_{d\in D_i, j\geq i} \text{died}_d(p_j)}$$

$$\frac{\sum_{d\in D_i, j\geq i} \text{died\_size}_d(p_j)}{\sum_{d\in D_i, j\geq i} \text{died}_d(p_j)}$$

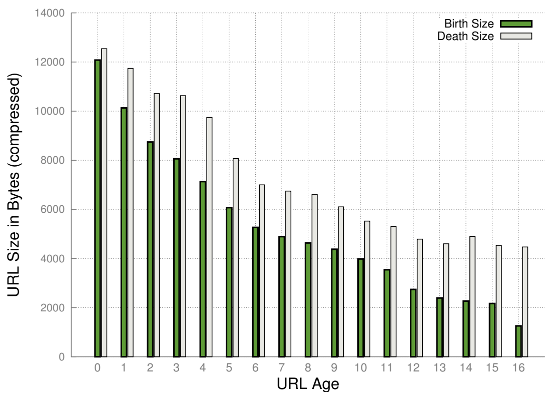


Figure 3.8. Average URL Birth/Death Size

Figure 3.8 shows these numbers in a cumulative manner, averaged over all URLs at a given age. Accordingly, URLs that die earlier tend to be larger than longer living ones. Hence, it appears that less content promises a longer lifetime. Furthermore, the plot shows that URLs grow in size over time, regardless of their age. This growth is almost constant, which indicates that longer living URLs either grow more slowly or that most of the growth takes place in the early years of a URL, as already found by Koehler et. al [52]. In contrast to that observation, short-living URLs with a lifetime of less than a year seem to grow least of all in size.

3.1.5 Archive Dataset Discussion

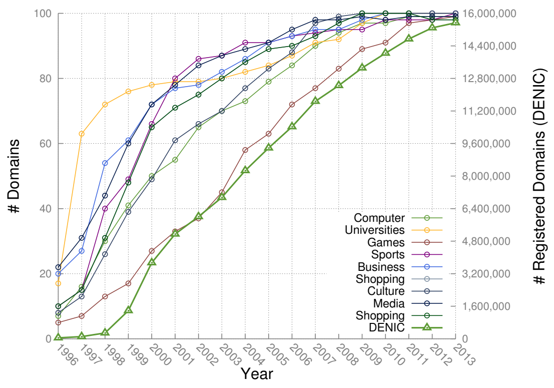


Figure 3.9. Domain Emergence vs. Registered Domains on DENIC (right y-axis)

Our analysis of Web evolution is performed on a dataset comprising German websites under the `.de` top-level domain, which was provided by the Internet Archive. The Internet Archive is the largest and most complete Web archive today. It covers a period of 18 years and constitutes a great source for analysis like ours. Just like in every other archive, not everything can be preserved. What is saved from the Web is influenced by crawl policies and constraints that impact both completeness and the change coverage.

We conducted this analysis under the major assumption that, if a domain is crawled, it is crawled completely with respect to the applied crawling policies and limitations, such as certain filters and maximum number of hops from a seed page. Hence, even though this does not cover all URLs of a domain, as long as the crawling strategy does not change over time, our observed trends are still valid. For Internet Archive crawls performed after 2010 this is actually the case. Thus, at least our results after that time are not affected by changing crawl policies at all. However, due to our focus on popular domains, we expect the assumption to be widely true also before 2010. The Internet Archive received lots of their crawls as donations from different partners. As crawlers, especially from search engines, typically aim for the most prominent part of the Web, we consider our subset consisting of popular domains to be covered with higher priority and hence very comprehensively compared to the rest of the archive.

Moreover, we investigated how well the analyzed popular domains in the Web archive represent the actual Web by comparing to the trend of registered domains on DENIC[5] (the `.de` domain registrar), as shown in Figure 3.9. The plot gives an overview of presence of domains from the different categories in our dataset at every year under consideration. A domain that is not present can mean two things: 1. it was not online at that time, or 2. it was not considered in the Internet Archive crawls. Although we are not able to distinguish this, the experiment shows a similar trend to the actually existing domains, suggesting that our dataset is fairly representative.

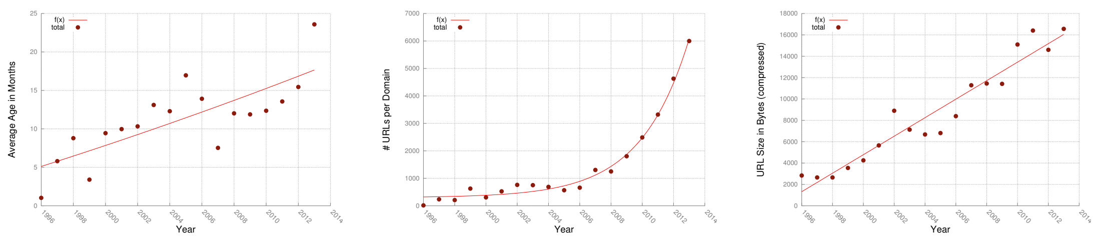
Interesting are also the differences among different categories: whereas about 75% of today's university websites already existed in 1999 and grew quickly, not even 20% of today's popular game websites were present back then. Most likely, many universities even had a website before 1996, but only got picked up by the crawlers later. By contrast the game websites that are most popular today have been created more recently and grown slowly since. The fact that perhaps not all domains were covered in the very beginning does not affect our analysis, as we investigated volume and size on a per-domain and per-URL basis, respectively.

3.1.6 Conclusion and Outlook

In this study, we have presented an extensive longitudinal analysis on 18 years of the popular German Web, based on crawls of the Internet Archive. We carried out an in-depth analysis on how the popular domains of today were created and how their age, volume and sizes have grown over the last decade. First, we find that most of the popular educational domains like universities have already existed for more than a decade. On the other hand, domains relating to shopping and games have emerged steadily over the period of the last decade. Second, we see that the Web is getting older, not in all its parts, but with many domains having a constant fraction of webpages that are more than five years old and aging further. Finally, we see that popular websites have been growing exponentially after their inception, doubling in volume every two years.

The study has provided us with interesting insights and ramifications on the evolution of the prominent part of the German Web. What we have learned about its growth and size can impact resource allocation strategies for Web archives as well as exhaustive and focused crawling strategies. Especially the identified differences among the studied categories can be of importance when dealing with topical or organizational Web archives from the respective areas. The introduced properties and definitions provide a solid foundation for comparing our findings on growth and aging against different Web archive collections. A possible research question would be: *How does the Web of other countries compare to this analysis of the German Web?* Furthermore, we lay the foundation for follow-up questions in future research, such as: *How do webpages evolve content-wise compared to size and age, and why is the average size of the newborn webpages today larger than*

[5] `http://www.denic.de/en/background/statistics.html`



(a) URL Age Evolution

(b) Domain Volume Evolution

(c) URL Birth Size Evolution

Figure 3.10. Predictions of Evolution Analysis

the ones in the yesteryear? Is it because of an actual increase in content or is it because of the markup due to constantly increasing Web authoring technologies?

3.2 Efficient Processing of Archival Collections

One of the fundamental tasks in using Web archives for research is *corpora building*. This task involves the selection and filtering of subsets, grouping and aggregation of records of interest and the extraction and derivation of new data (cf. Sec. 3.2.2). Consequently, there is a need for a framework that provides this functionality for efficiently constructing corpora out of the original archived collection. However, only providing fast access to the underlying collection is not sufficient. The framework needs to tackle a number of objectives driven by practical requirements (see Sec. 3.2.3), like simplicity, expressiveness, extensibility and the ability to produce reusable, well-structured output.

We address these objectives by proposing ArchiveSpark, a framework for distributed Web archive processing based on *Apache Spark* (see Sec. 3.2.4). ArchiveSpark is based on a novel two-step approach that first loads metadata as proxies of the corresponding data records and only reads the actual data records in a second step when required. By supporting standard file formats, we achieve the distinct advantage of institutions being able to easily share and apply the corpus generation specification across different collections. Towards providing efficient access, ArchiveSpark makes use of a metadata index (`CDX`) that is widely used by other tools in the domain of Web archiving. The `CDX` provides a lightweight representation comprised of metadata from all records in an archive. We achieve efficiency of access by exploiting the `CDX` to select records of interest before accessing the original archived content from the corresponding `(W)ARC`files. This random-access pattern of reading data was supported by the inclusion of data record pointers (*filename* with record *offset* and record *length*) in the `CDX` metadata records. Given that, we further laid out the goal to also support Web archive data from different locations, such as directly from the Wayback Machine without needing the records to be available on-site.

ArchiveSpark also delivers substantial speed-ups by using the *lightweight repre-*

sentations of records to enhance performance of distributed operations, like grouping and aggregation, unlike existing approaches that operate on much larger raw inputs. More specifically, rather than starting with all archived records and stripping them down, we operate on lightweight representation of records from the `CDX` and iteratively extend it as needed. We consequently observe large improvements in efficiency as we are able to minimize expensive disk operations involved as the researcher modifies and refines her requirements. To evaluate ArchiveSpark's performance, we compare and contrast our system with two alternative approaches and perform benchmarks to show differences in speed for select scenarios (see Sec. 3.2.5). The benchmarks show that ArchiveSpark is faster than a similar approach that does not make use of the metadata index in the selected scenarios, which we aim at. Also, depending on the task, ArchiveSpark is even faster than a method of filtering based on HBase, a distributed database system, without the space and time overhead of ingesting and storing the archived data into a database.

ArchiveSpark is open-source and contributions to extend its functionality are very much appreciated. For this reason, we provide convenient extension points and an architecture that makes it easy to apply third-party tools to create custom derivatives from Web archives as part of an ArchiveSpark job specification. The working tool with the functionality that we provide out-of-the-box is freely available under:

https://github.com/helgeho/ArchiveSpark

3.2.1 Related Work

Scientifically published articles on data extraction from Web archives, like ArchiveSpark, have been very limited. To the best of our knowledge, the only comparable system is Warcbase, later renamed the Archives Unleashed Toolkit (AUT)[6], by Lin et al. [62], which serves as the baseline in our benchmarking process (see Sec. 3.2.5). There are also a number of other approaches in the area of accessing and mining Web archives including tools from industry.

In this discussion on related work we differentiate between specialized Web archive access approaches based on certain properties and more general approaches. The formers provide search and lookup operations as the method of access, while the latter provide access to all the archived data with support for data processing. ArchiveSpark, our tool for general Web archive access, supports arbitrary filtering and data derivation operations on archived data making it much more suitable for the scientific use of Web archives. Furthermore, ArchiveSpark's concept of *data specifications* make it a universal tool for *distant reading* of archival collections, which will be discussed in the end of this section.

[6]https://github.com/archivesunleashed/aut

Specialized Web Archive Access

The Internet Archive, as one of the driving institutions of Web archiving, and most other Web archives, feature the Wayback Machine or *Open Wayback*[7] to provide access to their Web collections. The Wayback Machine enables URL based access to the archived captures of a website, based on a server API powered by a metadata index (`CDX`). Lookups are designed for efficient, random URL based access and accomplished by running binary searches through the sorted index files. Researchers can query the `CDX` server for metadata information of a particular URL, host, domain or URL prefix.

In contrast to these structured queries by means of metadata, the *UK Web Archive*[8] is working on an information retrieval system based on the Apache Solr search platform[9]. Their *Shine* project[10] supports faceted searching and more sophisticated trend analysis of Web archive content. Hockx-Yu [30] identifies 15 Web archives that feature similar kinds of full-text search capabilities. While these are largely engineering efforts that exploit existing search systems, there have also been scientific efforts to build indexes specialized on certain properties, such as time [42] or semantic annotations [114]. There are however two major challenges with these approaches that limit their applicability in the area of corpus building from Web archives. First, it is not always feasible to obtain the necessary computational resources to parse and index all archived Web content and store them in a search index. Second, even if the necessary resources are available, they cannot efficiently support corpus building processes that go beyond these specialized lookups. For these reasons, with ArchiveSpark, we propose a general data processing approach that exploits the `CDX` for gains in efficiency while not having to rely on an external index.

General Web Archive Access

Due to the size of Web archives, often in the order of multiple terabytes, a single machine can no longer process or even store those collections. As a result, distributed computing facilities are commonly implemented for processing archived data. In contrast to the previously discussed specialized access approaches, these facilities enable general access to the archives by operating directly on the data records for selection, filtering, aggregation and transformation.

As part of their self-guided workshops, like the *Web Archive Analysis Workshop*[11] and *ARS Workshop* [12], the Internet Archive provides a number of tools for this purpose. These tools enable researchers to batch process data and derive information like hyperlink graphs and mined text using, *Apache Hadoop*, an open-source

[7] `https://github.com/iipc/openwayback`
[8] `http://www.webarchive.org.uk/`
[9] `https://github.com/ukwa/webarchive-discovery/wiki`
[10] `https://github.com/ukwa/shine/wiki`
[11] `https://webarchive.jira.com/wiki/display/Iresearch/Web+Archive+Analysis+Workshop`
[12] `https://github.com/vinaygoel/ars-workshop`

implementation of the MapReduce programming model for distributed computing of large datasets [58].

AlSum [60] presents with *ArcContent* a tool for archive access based on Hadoop that uses Cassandra [61], a distributed database to store the extracted data. Similar to ArchiveSpark, it involves a data filtering step where records of interest are selected using the metadata fields in the corresponding `CDX` dataset. However, in contrast to our approach, the extracted records are stored into Cassandra to be queried through APIs powered by a Web service. This only works in cases where the research task is clear and well-defined beforehand and does not involve iterative filtering and data transformations.

Most similar to our ArchiveSpark framework is Warcbase by Lin et al. [62], later renamed the Archives Unleashed Toolkit (AUT). This open-source platform for data processing on Web archives provides two different methods to access the data and served as a baseline in our benchmarks (see Sec. 3.2.5). Warcbase was originally developed to be based on *HBase*, an open-source implementation of Google's Bigtable [63], a Hadoop-based distributed database system. It features tools to ingest the Web archive records into HBase and allows for temporal browsing of URLs, with efficient, random URL based access similar to the Wayback Machine. The first method requires the storing of data in HBase with researchers leveraging Hadoop based tools to analyze it. However, this has the major drawback of involving an expensive setup phase of duplicating the entire Web archive in HBase. For the second method, Warcbase provides convenience functions to load and process the archive files directly using *Apache Spark*, one of the most popular alternatives to Hadoop. Spark, in contrast to Hadoop/MapReduce makes extensive use of the main memory of nodes, which has shown to lead to impressive speed-ups [59]. With the name change to the Archives Unleashed Toolkit, the only supported method now is the one based on Spark in order to avoid the previous HBase overhead of Warcbase [115]. However, this Spark based method, in contrast to ArchiveSpark which is also based on Spark, does not optimize for efficiency or meet all the objectives outlined below.

Universal Distant Reading

ArchiveSpark's modular architecture allows to implement any logic to load metadata and data, from any source and of any type, even beyond Web archives, like digitized journals and boos. This generalized support makes it a universal tool for *distant reading* and to the best of our knowledge, it is the first one of its kind. Distant reading refers to the technique of analyzing large corpora of text documents without closely reading every single one. Schulz [116] describes it as *"understanding literature not by studying particular texts, but by aggregating and analyzing massive amounts of data"*. The idea was proposed by Moretti [117] as a way to analyze texts systematically, using statistical and quantitative methods on texts, which are often expressed as networks of terms, where the edges represent the relationships among the terms. While it was originally meant to analyze literary fiction, we conceive it as a more general tool to derive information from big collections without reading single documents. A good overview of the related works in distant reading from the visual analytics perspective for *Digital Humanities* has been published by Jänicke et al. [118].

An area that is closely related to this, but in which distant reading and archival collection are often neglected, is *Big Data*. In that field, Maemura et al. [119] presented a framework to document the research process in Web archives, contributing to a better understanding of the findings and their provenance, in order to reuse of data, methods, and workflows. ArchiveSpark provides a technical solution to this by enabling a rather declarative description of data processing workflows that can be reused among different datasets, even beyond Web archives. This supports new processes to work with Big Data across teams and projects in coordination as well as data sharing, as sought by Saltz [120]. At the same time, we tackle interoperability challenges, widely present when dealing with Big Data [121].

3.2.2 Use Case Scenario

In order to use Web archives as a scholarly source for scientific research, a required first step in most cases is the extraction of a well-defined corpus to work with [30, 122]. Scientists typically focus on a temporal and/or a topical subset of the archived data within the scope of their research question. In the following example, we consider five steps to be taken by a political scientist who wants to analyze sentiments and reactions on the Web from a previous election cycle.

Step 1: The researcher would need to define and extract a specific Web collection related to her research. In this case, she would only need websites that were archived in the time period of interest. However, this time-based or longitudinal filter alone would result in too many candidate websites as most Web archives are not topically organized. Finding just election related websites from this candidate pool requires domain expertise and/or manual intervention. For that reason, it is useful to have this pool to be as small as possible to begin with.

Step 2: Since the researcher needs to consider only text resources from websites for her sentiment analysis project, she would apply a filter on *MIME types* to only select such resources. However, identifying these resources by their MIME type involves accessing and parsing the HTTP headers of the records in the archive, which is a low-level detail and needs to be abstracted away from the potentially non-technical researcher.

Step 3: Another required filter involves the *HTTP status code* of a particular capture. The fact that a certain URL was captured at some point in time and is part of the archive does not necessarily mean there was a valid Web resource being served at the URL. The URL could have been the result of an invalid link or a dead URL that was valid at a previous time. As our example researcher would only be interested in successful URL fetches, she would need to filter for records with status code 200.

Step 4: At this point, the candidate set is still likely to be very large for manual analysis. The researcher might decide to only focus on websites that contain certain terms or a specific set of entities, e.g., the candidates of the election. While seeming straightforward, this content-based filter involves accessing the content of every candidate record, which in turn involves separating the headers from the response body, encoding the textual response to a string, parsing out raw text from HTML, and finally applying text processing tools, before filtering on the resulting values.

Step 5: Web archives typically contain multiple captures of a website for every time the website was crawled, regardless of whether it has changed or not. Therefore, our researcher might decide to pick only the latest captures of the candidate URLs. In order to apply this filter, all captures of the intermediate corpus need to be grouped by URLs and sorted by their capture times. These types of operations are very expensive when performed on the raw records that include the entire payload. By operating only on metadata records that contain the required fields, they can be made much more efficient. However, this implementation is not something that the researcher should necessarily be concerned with.

Filtering, selection, grouping and extraction steps, like the ones described, can be arbitrarily continued. Depending on the task at hand, it may be necessary to keep track of where a certain value was derived from. As an example, consider the case that the researcher deems entities extracted from the title text to have more value than those extracted from the body text on a page. Keeping track of this lineage is an essential way to document the collection building and derivation process and enable its comprehension and reproduction by other researchers. Therefore, it should be included in the output format to be used by the researcher in her further research process.

ArchiveSpark seeks to tackle the challenges that arise by a research scenario such as the one described above. A researcher or a technical person supporting her on the corpus building process should be able to easily specify her requirements and efficiently extract the required corpus from a Web archive.

3.2.3 Objectives

ArchiveSpark addresses six objectives, which we identified as being essential for a tool for corpus creation on Web archives, based on practical requirements. These comprise (1) a simple and expressive interface, (2) compliance to and reuse of the standard formats in the domain of Web archives, (3) an efficient selection and filtering process, (4) an easily extensible architecture to support various derivation tools, (5) lineage support to comprehend and reconstruct the process of derivation from the archive, and (6) an output in a standard, readable and reusable format.

Simple and Expressive Interface

The primary objective, when we designed ArchiveSpark, was a simple interface that lets users access the fields of interest without the need to do any parsing of archived Web records themselves. Users of this interface would be able to easily express any selection and filtering operations and access available information without carrying over complete archived records at each stage of the workflow. Additionally, the idea was to provide a seamless transition from filtering based on just metadata available in the index to that based on the contents of the archive.

Since ArchiveSpark is based on Spark, which is written in Scala, we naturally chose Scala to be the language of choice for ArchiveSpark. Scala enabled us to specify the ArchiveSpark extraction and derivation workflow in a functional manner. This functional approach is less verbose than that of traditional object-oriented languages and often simplifies tasks as it allows for a more natural way of expressing thoughts. Our interface is inspired by the existing Spark API and the Scala standard library, to provide the same degree of simplicity and expressiveness in a rather declarative manner.

Even though the interface, in our opinion, is fairly intuitive to use by a computer scientist or a researcher familiar with programming, we do not expect researchers from other disciplines to be able to use it directly in all cases. However, with the aid of a technically savvy person, the researcher should be able to express her thoughts and requirements on the collection building process and get them easily translated into an executable ArchiveSpark workflow.

Standard Formats

In the area of Web archiving, there are a couple of file formats for storing archived Web resources and derived metadata that have been established and in wide use over the years. As a result, these formats have either become de-facto standards or have been standardized by ISO. Given their common availability in almost every known Web archive, we wanted our system to be based on these file formats. We did not want to introduce any new file format or index structure: while such files or indexes could provide gains in efficiency for access, their generation would necessitate a pre-processing phase consuming expensive compute resources and additional storage. While being based purely on pre-existing file formats, ArchiveSpark maintains its essential objective of efficiency as described in the next sub-section.

The most important format in the world of Web archives is `WARC` (Web ARChive), which is registered as ISO 28500. `WARC` is a format to store archived Web resources. Every record in a `WARC` file represents the capture of a single Web resource at a given instant of time. The `WARC` record comprises a header section that includes the URL of the resource, the timestamp of capture and other metadata, as well as a payload section that contains the body returned by the Web server. In the case of HTTP responses, the payload consists of a HTTP header and body. Before `WARC` was introduced as a format to store Web archives, archived records were widely stored in the older `ARC` format[13]. Although `ARC` is not standardized, many Web archives still contain data in this format, and hence ArchiveSpark supports both `WARC` and `ARC` file formats.

Another format which is not standardized but is seen as a de-facto standard is `CDX`[14]. This is an index format that contains a number of metadata fields for every

[13]http://archive.org/web/researcher/ArcFileFormat.php
[14]http://archive.org/web/researcher/cdx_file_format.php

Web capture including pointers to the `(W)ARC` file and the file-offset into the file where the capture is stored. A header line specifies the metadata fields contains in the plain text index file. Most commonly generated, however, are `CDX` files with either 9 or 11 fields, which are utilized by the Wayback Machine to serve records to users browsing the archive. Since the Wayback Machine software is currently the access method of choice for most Web archives, `CDX` files are generated by and/or readily available to these archives. As an example, `CDX` files are available for the crawls provided by the Common Crawl initiative[15]. Furthermore, it is possible to generate both `WARC` and `CDX` files with the current version of the Unix/GNU download tool *Wget*[16].

In summary, with ArchiveSpark we designed for, first, being compliant to these standard formats, and second, not introducing and depending on any new format. This way we aim to guarantee that any Web archiving institution that has `(W)ARC` and corresponding `CDX` files can use ArchiveSpark to extract and mine their Web collections, without requiring any expensive pre-processing steps or prerequisites.

Efficiency

Efficiency is one of the core objectives of ArchiveSpark. Since Web archives are typically large-scale data collections of terabytes or even petabytes, a scan-based selection over all archive files is a very time-consuming process and can potentially run in the order of multiple days. This is in most cases too inefficient to be used for corpus building as part of a scientific research task.

With ArchiveSpark we leverage the available `CDX` index files. As a first step, we apply filters on the metadata fields from the `CDX` and generate a small candidate pool with the captures of interest that need to be read in from `(W)ARC` files. This way, we potentially avoid the scenario of reading in all the records in the archive before ending up rejecting a large number of them (see Sec. 3.2.4). Our approach of `CDX`-enabled filtering and selective data access results in efficiency gains over the scan-based approach.

Furthermore, when working with the raw archive records, complex operations, like groupings and aggregations, become much more expensive, since the whole records need to be moved around in a distributed setting. This could be optimized by stripping out data that is not required by those operations. However, if needed later, it will need to be recovered from disk, which is often even more expensive.

With ArchiveSpark we turned this around using a selective data access and derivation approach, starting with lightweight records consisting of only metadata and iteratively extending them as needed, resulting in further gains in efficiency.

[15]`http://blog.commoncrawl.org/2015/04/announcing-the-common-crawl-index`
[16]`https://www.gnu.org/software/wget`

Extensibility

In most research applications, instead of working on the raw archived resources, a researcher is interested in extracting or deriving the data of interest for a given research task. Derivations can either be created from the original payload of an archived resource or from previously derived data. An example of such successive derivations on text are Natural Language Processing (NLP) tasks, such as the extraction of named entities from websites. The corresponding derivation tools operate on natural text and thus, first require the HTML parsers to remove markup and extract plain text, followed by the NLP tool, i.e., the named entity extractor, to extract the desired information.

There are a limitless number of other derivations that researchers can be interested in, e.g., audio/video fingerprinting on archived media files, OCR on archived images and many others. With ArchiveSpark we want to ensure any possible derivation from Web captures, regardless of whether they were constructed by us beforehand or not. Therefore, we designed a very flexible architecture with appropriate extension points that allow the application of custom code as well as third-party libraries to build derivatives from the records of a researcher corpus.

Traceability

An important trait of any scholarly resource is transparency and traceability. In order to make scientific research reproducible it is essential to understand how the research corpus was designed. However, in the case of Web archives, it is difficult to retrospectively reproduce the crawling process. Reasons for this are, among others, an ever-changing Web, a semi-automatic prioritization by Web crawlers, changing crawling strategies as well as multiple, disparate parties being involved in the collection process. As a result, we found it even more important to focus on documenting the data lineage of corpus building from Web archives.

Also, depending on the needs of the researcher, it may often be sufficient to only deal with derived information and not include the original records. In order to reproduce this derivation process at a later time, a proper documentation of the data lineage is absolutely crucial. ArchiveSpark achieves this objective of traceability by documenting the data lineage of all the derived records. The documentation includes metadata that allows for the identification of all the source records responsible for the derivative as well as the derivation path outlining the steps undertaken to filter, transform and derive from these records.

Reusable Output

The extraction and derivations steps performed by ArchiveSpark act as a preprocessing phase in a research pipeline. The data extracted from the Web archive serves as scholarly source for research tasks, which can either be manual or programmatic. In the case of manual research, researchers would typically create rather small, very selective corpora and read in the results manually. On the other hand, researchers may use tools to analyze the corpora based on different features in a completely automatic or semi-automatic manner.

In either case, the corpus needs to be clean, well-structured and readable. While human readability implies a pretty printed output without too much clutter, machine readability implies data parsing support. The latter can be guaranteed best by producing data in a commonly used format with existing parsers for various programming languages. One such format is JSON, which was originally introduced as an exchange format for JavaScript to be used by Web services. However, because of its simplicity, it has become a widely used format that can be easily parsed by many pre-existing tools.

JSON supports a cascading nested structure with multiple levels of data and is therefore well-suited for supporting the data lineage functionality of ArchiveSpark as described above. Another advantage is that these nested cascades of data can be easily presented in a fairly human readable form. For these reasons, we decided on JSON as the default output format of choice. Of course, any other output format that meets our outlined objectives can also be implemented and integrated into ArchiveSpark.

It is worth noting that the use of ArchiveSpark is not restricted to such an output. Researchers can also use it to access the archive, apply filters and derivations, and continue using the rich data types provided by ArchiveSpark in a Spark job to perform data analysis at scale, e.g., machine learning or graph analysis.

3.2.4 ArchiveSpark Concepts and Architecture

ArchiveSpark is a framework that enables efficient data access, extraction and derivation on Web archive data with a simple API that enables flexible and expressive queries. The following sections describe the approach as well as the distinct features of ArchiveSpark, designed to meet the previously described objectives.

Approach

ArchiveSpark makes use of a metadata index to selectively access resources from a Web archive. This approach is optimized for efficiency when extracting a defined subset of records as it avoids having to perform a full scan through all records in an archive. Since corpora used in scientific fields typically consist of data derived from a small subset of the entire Web archive, ArchiveSpark is well suited for these use cases.

Figure 3.11 shows how ArchiveSpark works in the standard case of metadata being loaded from CDX and the corresponding records stored in (W)ARC files. First, the filtering process is performed using only metadata contained in the CDX files. Second, by utilizing the file pointers contained in the CDX records, ArchiveSpark selectively accesses the filtered records from the underlying (W)ARC files. At this stage, we augment the record's metadata with headers and content from the (W)ARC records. Next, users apply what we term *enrichments* to derive new information,

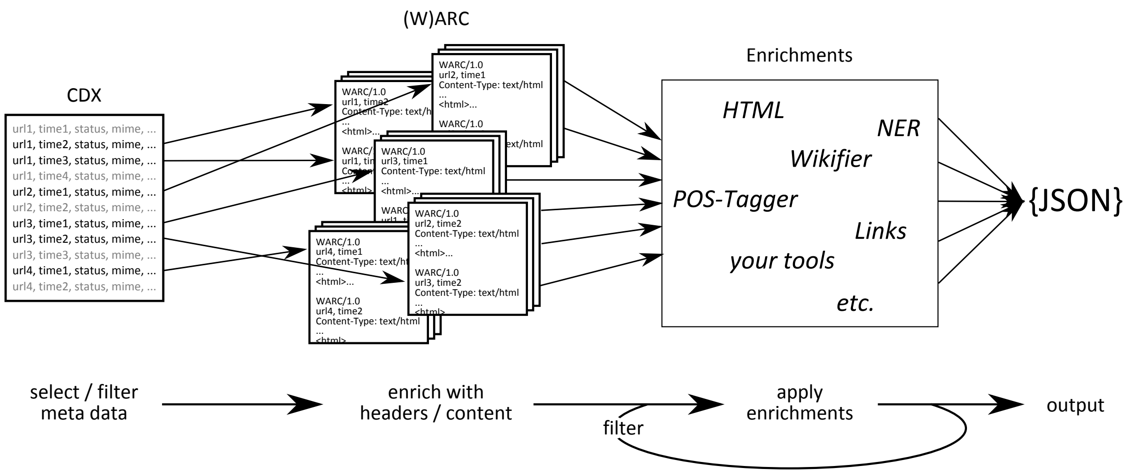


Figure 3.11. Illustration of the ArchiveSpark Selection and Enrichment Approach

such as named entities or hyperlink data, that is added to the records. These enrichments can be applied by executing custom code or external tools. Based on the derived information, further filters and enrichments may be applied iteratively. The resulting corpus can be saved in a custom `JSON` format that is tailored to support data lineage.

Modularity

ArchiveSpark's flexibility of supporting various data sources and types is achieved by the concept of *data specifications* (`DataSpecs`). A `DataSpec` defines how metadata records are loaded and how they are associated to the corresponding data items. It encapsulates the code to load and access metadata as well as corresponding data records, while the details are abstracted away from the users. These isolated objects allow for easy sharing of datasets as well as data processing pipelines in the form of recipes, which are defined in a clean, declarative manner with the associated complex logic defined separately. At the same time, a given `DataSpec` is fully customizable by its developer and may be parameterized, which allows the user to easily specify required information, such as a data path or other types of location pointers. Figure 3.12 illustrates how `DataSpecs` fit into ArchiveSpark's architecture and play together with the other components.

Once ArchiveSpark is instructed to load a given `DataSpec`, the corresponding dataset is presented to the user as a collection of `EnrichRoot` records, the entry points into ArchiveSpark's data model. The initialization of such records is part of the used `DataSpec`. For instance, a common `DataSpec` for Web archives creates records that hold the corresponding `CDX` record information, while one for book records would contain other kinds of metadata, like a book's title and author. This ability to support other types of archival collections will be addressed in Section 3.2.6. The specialized `EnrichRoot` records then provide access to the actual data as defined by the used `DataSpec`.

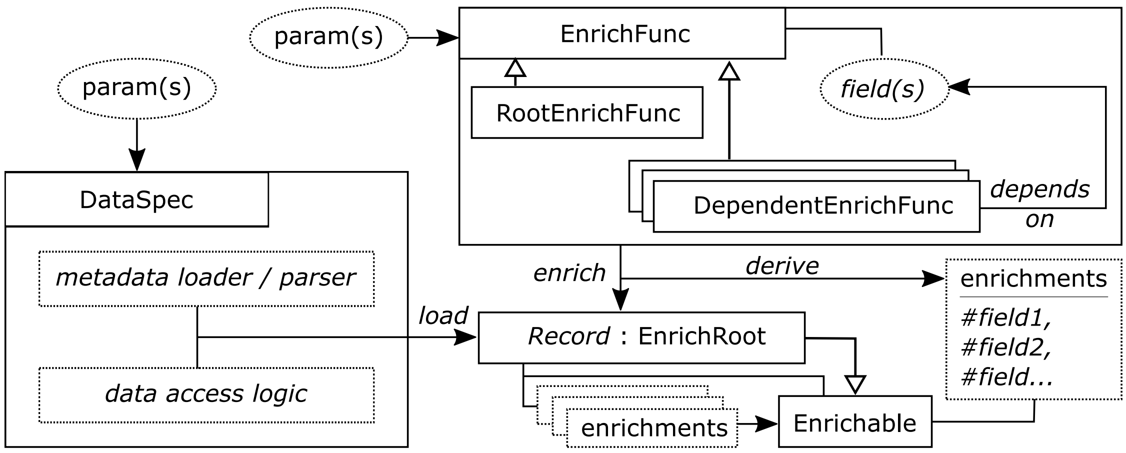


Figure 3.12. ArchiveSpark Architecture

Starting from there, ArchiveSpark's data model constitutes a *hierarchical tree structure* with `EnrichRoot` as the root node. Each node in this tree model is of type `Enrichable` and holds a value (for the root node, this is typically the metadata record), as well as zero or more child nodes with the child nodes representing either extracted or derived data based on the parent's value. Hence, the hierarchy of the data encodes its lineage, which corresponds to the dependency hierarchy of the applied functions.

Enrich functions (`EnrichFunc`) describe transformations that can be applied to the values stored in ArchiveSpark's data model. They encapsulate arbitrary logics and even third-party tools. Similar to a `DataSpec`, an `EnrichFunc` is used as an isolated black box in a declarative manner and can be configured by the user. All enrich functions, except for the root, have a dependency that determines their input, i.e., the dependency function's output will be the input of a `DependentEnrichFunc`. The very first `RootEnrichFunc` in such a dependency chain is in charge of loading the data of a record with accesses provided by the `EnrichRoot`.

To reuse and share an `EnrichFunc`, users can easily change its dependencies. As an example, consider a function that has been defined to depend on the body text of a webpage, i.e., its direct dependency is the `EnrichFunc` that extracts this text from the webpage. By changing this dependency dynamically to the function that extracts the title of a webpage, the same `EnrichFunc` will now operate on the title as input. Similarly, by changing the root dependency, a `EnrichFunc` can now by universally applied, regardless of the input data type, i.e., a function that has been developed to extract entities from text is now applicable to any text, regardless of whether it is text on a webpage or in a book.

Currently, we provide the most basic enrich functions to get users started, but we will continue to extend ArchiveSpark with more functions moving forward. As ArchiveSpark is fully open source, anyone can also contribute to its development and provide their own tools as enrich functions in separate projects. To support this, we provide convenient base functions that make it easy for a developer to define custom enrich functions.

Interface

The interface of ArchiveSpark lets users define the extraction and derivation workflow in a declarative manner. It is based on Apache Spark and greatly inspired by its API (Application Programming Interface). Also, ArchiveSpark uses the data structures of Spark and is hence fully compatible with any transformation methods provided by Spark. Like Spark, ArchiveSpark is implemented in Scala, a functional and object-oriented programming language running inside the JVM, Java's runtime environment. As a result, it is compatible with any third-party library running on the JVM as well, for instance all available Java and Scala libraries.

The entry point to ArchiveSpark is a globally available object with the same name. It serves as a starting point by providing methods to load the data into so-called Spark RDDs (Resilient Distributed Datasets). RDDs are partitioned collections of objects spread across a cluster, stored in memory or on disk. Spark programs are written in terms of operations on RDDs.

Through `DataSpec`'s as described above, we support various combinations of metadata and record types as well as sources. For instance, in order to load Web archives from local `(W)ARC` and `CDX` files, stored in Hadoop HDFS (Hadoop Distributed File System), we would use the provided `WarcCdxHdfsSpec`, which expects the both file locations as parameters. The following code is written in Scala, since it is our language of choice for defining an ArchiveSpark workflow specification:

```
val archive = ArchiveSpark.load(
        WarcCdxHdfsSpec("path/to/CDX", "/path/to/(W)ARC"))
```

The above *archive* variable now references a Spark RDD consisting of specialized ArchiveSpark records. Hence, all methods provided by Spark to manipulate it through a set of parallel transformations, e.g., filter, as well as actions, e.g., count, can be applied. However, at this point these are based on the `CDX` data and therefore, only allow access to the metadata fields available in the `CDX`.

The following call applies filters on HTTP status codes and MIME types and only retains those records with a successful response (*HTTP status code 200*) of type *text/html*:

```
val filtered = archive.filter(r =>
        r.status == 200 && r.mime == "text/html")
```

In the functional paradigm of Scala, every operation returns a new, immutable object instead of modifying the previous one. We have made sure this behavior is provided by ArchiveSpark as well. Hence, *archive* still represents the entire dataset, while *filtered* is a new object representing the filtered one.

As all Spark transformation operations are lazily evaluated, no actual data access will have been performed yet. The original RDD as well as the filtered one are just representations of the corpus to be extracted from the Web archive. The above filter is only evaluated or executed once a Spark *action*, such as a data output, is performed. The advantage of the lazy loading is that, although all `CDX` records need to be read, only those that have passed through the filters are kept in the dataset consuming much less memory.

To access the actual content of these records in the next step, ArchiveSpark provides a method to apply so-called *enrich functions*. The most basic enrich function is *WarcPayload*. It opens the `(W)ARC` records, which are pointed to by the selected `CDX` records in the dataset, parses the HTTP response and enriches the original records with four fields: 1. `(W)ARC` header, 2. HTTP status line, 3. HTTP header, and 4. Payload:

```
val response = filtered.enrich(WarcPayload)
```

Enrich functions can depend on each other and be applied consecutively. Each consecutive application derives new information from its parent dependency. While *WarcPayload* does not depend on any other enrich function and is usually applied first, *StringContent* depends on *WarcPayload*. It transforms the payload of every record in the dataset into a string representation and enriches the record with this string. This works because our filter on the MIME types before made sure that our example dataset only contained text responses and no images or other binary files:

```
val strings = response.enrich(StringContent)
```

By explicitly enriching the records with both *WarcPayload* and *StringContent*, ArchiveSpark marks both these fields to be contained in the output. This way, by specifying what the records should be enriched with, the researcher can control the required features in the final corpus. If the dataset referenced by the *response* variable had been directly enriched with *StringContent*, only this enrichment would have been part of the output. However, internally, this process would still have first enriched the dataset with *WarcPayload* as it is dependent on the payload. And since the payload was already present in the records from an earlier enrichment, the payload would have been used as-is and would not have needed to be re-computed. Note that dependencies specified in enrich functions are defaults but can also be explicitly specified by the user. For the sake of clarity and brevity, we do not show all the currently available methods and options of ArchiveSpark here.

Based on the enriched information, additional filters can be applied. This process of enriching and filtering can be repeated as needed. For the most efficient execution, it is recommended to apply filters as early as possible i.e. as soon as the data to be filtered on is available. This guarantees that any expensive derivation is performed on as few records as needed. This is especially important for the very first enrichment operation, which involves accessing data from `(W)ARC` files.

Other than the metadata fields available from the `CDX` records, the data derived by enrich functions is not typed, as different functions can create fields of various data types. The access to these values is enabled by specifying a path in dot-notation, where each segment specifies a level in the derivation pipeline. To facilitate this access, the corresponding enrich functions encode these paths and can be used instead, e.g., `StringContent` points to `payload.string`. ArchiveSpark provides getter methods that support both ways of referring to a value. As an example, the following instruction filters on the content string, i.e., the HTML code in the case of a webpage, and retains only those records that include the term *internet*:

```
val internet = strings. filter (
        r => r.getValue(StringContent).contains("internet"))
```

After the final dataset has been created, it can be written out as JSON using the *saveAsJson* method on archive records RDDs provided by ArchiveSpark. It transforms the records into JSON objects consisting of the metadata and all explicitly enriched data:

```
internet .saveAsJson("/output/path/results.json.gz")
```

The *gz* extension is automatically detected by ArchiveSpark and causes it to compress the output using *gzip*. The above six instructions have now created a corpus consisting of all successful text/html responses, i.e., HTML webpages, that contain the term *internet*, formatted as pretty-printed and well-structured JSON in a compressed form.

As an alternative to the JSON output, users are free to transform the archive records that ArchiveSpark uses as its first-class citizen into any form they want. We provide all the necessary access methods for this purpose. That way, besides the corpus building use case, ArchiveSpark can be used as a library to access Web archives as part of a larger data analysis application pipeline.

Formats and Lineage Support

With the data specifications provided as part ArchiveSpark's core codebase, we support one of the most common CDX formats that is in use by the Internet Archive's Wayback Machine. This format encodes eight metadata fields and three additional fields pointing to the (W)ARC file where the capture is stored along with file-offset and compressed length of the record. However, we can easily support additional CDX metadata fields as well as other types of metadata and data records, simply by creating new DataSpecs, which can be shared with other users in separate projects.

The default CDX is a space-separated plain text format with each line representing one record. A single header line at the top of a CDX file denotes the fields: *SURT URL (Sort-friendly URI Reordering Transform)*, *timestamp*, *original URL*, *MIME type*, *HTTP status code*, *content digest/SHA-1 checksum*, *redirect URL* (or -), *meta tags* (or -), *(W)ARC record compressed length*, *(W)ARC record file-offset*, *(W)ARC filename*.

An example CDX line looks as follows:

```
com,example)/jcdl 20160117113253 http://example.com/jcdl text/html
200 RKMS6XLYED4G8POFQUIN37WDEWYLD9Z - - 12345 67890 archive.warc.gz
```

For the default output format, we decided on JSON, a widely used format that meets our objective of a reusable output and in addition to that nicely suits ArchiveSpark's tree-like data model. Each output JSON record includes a listing of all the metadata fields from the source CDX identifying the selected resource. If no enrichments are applied, this would be the final output for our example record:

```
{
  "record": {
    "surtUrl": "com,example)/jcdl",
    "timestamp": "2016-01-17T11:32:53.000+01:00",
    "originalUrl": "http://example.com/jcdl",
    "mime": "text/html",
    "status": 200,
    "digest": "RKMS6XLYED4G8POFQUIN37WDEWYLD9Z",
    "redirectUrl": "-",
    "meta": "-"
  }
}
```

Enrichments are added to these JSON objects as additional keys next to *record*. In case the *WarcPayload* enrich function is applied, as in our example above, the (W)ARC headers, HTTP status information, HTTP headers as well as the raw bytes of the payload will be added in:

```
{
  "record": {...},
  "recordHeader": {
    "subject-uri": "http://www.example.com/",
    "content-type": "text/html",
    "creation-date": "20160117113253",
    ...
  },
  "httpStatusLine": "HTTP/1.1 200 OK",
  "httpHeader": {
    "Date":"Sun, 17 Jan 2016 10:32:53 GMT",
    "Connection":"close",
    "Content-Type":"text/html",
    ...
  },
  "payload": "bytes(length: 2345)"
}
```

Any other enrich function that depends on a value produced by *WarcPayload*, e.g., payload, will result in the output being added as a nested value. If, for instance, the dataset was enriched with *StringContent* only, which calls *WarcPayload* implicitly as its dependency, the resulting JSON might look like this:

```
{
  "record": {...},
  "payload": {
    "string": "<html>...</html>"
  }
}
```

In this case, the record and HTTP headers are not included, since the user did not explicitly specify them to be part of the corpus. The payload, however, is required to document the lineage of the string (the string representation of the payload). This meets the traceability objective of Section 3.2.3 as every derived value can be traced back through the cascades to its origin.

When the user is interested in both the original value as well its derivations, for instance, when the records in the dataset are additionally enriched with the length of their content string, a special underscore key (_) is introduced. The field with this key retains the original value, like in the following example:

```
{
  "record": {...},
  "payload": {
    "string": {
      "_": "<html>...</html>",
      "length": 2345
    }
  }
}
```

Other derivatives based on this string content would be placed next to the underscore, just like *length*. In the same way, if the dataset was explicitly enriched with both *WarcPayload* and *StringContent*, the byte representation of the payload along with the header fields would have been placed next to *string*.

Finally, we consider the example of deriving named entities from the titles using the HTML string representation. This example would involve an HTML parser, which depends on StringContent to enrich the dataset with the required title value nested under an HTML field, as well as a named entity extractor tool, which in turn depends on the title to create a set of named entities. The lineage path of this constructed example would look as follows:
payload.string.html.title.entities.

Remote Access

Web archives are often enormous collections with sizes in the order of hundreds of terabytes and not every research institution has the capacity or the infrastructure to maintain their own local Web archive. However, there are institutions, such as the Internet Archive, that provide public access to their holdings. ArchiveSpark's modular concept of data specifications that encapsulate the logic to load metadata and data records includes the ability to do this not only from local disk. Using different protocols, such as HTTP, it also enables to load data from remote sources like the Wayback Machine. To obtain the required metadata, queries can be made to the Internet Archive's *CDX server*[17], which provides access to the metadata of every capture in the Web archive. For this particular case, we provide the `WaybackSpec` with ArchiveSpark. Instead of paths for `CDX` and `(W)ARC` files, this spec accepts a URL query along with a few other settings of the `CDX` server, e.g.:

```
WaybackSpec(url = "http://example.com/", matchPrefix = true)
```

An advantage of querying such a service to load metadata records is the ability to prefilter records based on criteria supported by the query service. We note that, although the retrieved metadata records from the `CDX` server do not include explicit location information for the corresponding `(W)ARC` records, access to the data records is possible through the Wayback Machine by using the unique tuple of *URL* and *timestamp*. Similarly, other services can be incorporated if these values are provided by them. Another example of this is retrieval systems like Tempas to pre-filter records by keyword, which is demoed in [6] and used later in Chapter 4. By not requiring an ArchiveSpark user to possess a local copy of the data, we open up broader opportunities for data processing by multiple parties.

3.2.5 Benchmarks

We ran benchmarks to assess the efficiency benefits of exploiting the `CDX` dataset when accessing Web archives. The run times of three different scenarios are compared using ArchiveSpark and two baseline approaches: a scan-based approach using pure Spark, and the Warcbase approach using HBase. For both baselines, we used the tools provided by Warcbase to load and access the datasets (see Sec. 3.2.1).

Dataset

One of the services provided by the Internet Archive is Archive-It[18]. It is subscription based and enables partner institutions to run selective focused crawls to create and archive their own thematic and event driven collections. For our experiments, we chose one of these collections, the *Occupy Movement 2011/2012*[19] collection, collected by the Internet Archive itself. Unlike a generic Web crawl collection, this collection features a well-defined scope and is not too large, allowing our benchmarks to be performed in a reasonable amount of time.

The collection contains a total of 17,478,067 (17.4 Million) captures with 10,089,668 (10.08 Million) unique URLs. It contains Web content crawled during the time period Dec 3, 2011 to Oct 9, 2012, with a total storage of 470.9 GB of compressed `WARC` files. The `CDX` data, generated by us, adds in 24.4 GB of data size.

[17] https://github.com/internetarchive/wayback/tree/master/wayback-cdx-server
[18] https://archive-it.org
[19] https://archive-it.org/collections/2950

Experimental Setup

The experiments were performed on a Hadoop cluster running the Cloudera distribution[20] (Hadoop 2.6.0-cdh5.4.9). The cluster consisted of 2 master nodes and 24 compute nodes with a total of 256 CPU cores, 2560 GB of RAM and 960 TB of hard disk space.

The three systems we compared in the benchmarks were:

1. ArchiveSpark
2. Spark: Using Warcbase's Spark library
3. HBase: Using Warcbase's ingestion tool

For both ArchiveSpark and pure Spark approach, `WARC` files from the collection were stored in Hadoop HDFS. The `CDX` files required by ArchiveSpark were generated using the Internet Archive's `CDX` generator, which is available open source on GitHub[21]. Generating the `CDX` files took 110 minutes, however, this is a one-time process and is anyway a necessary step to enable access services like the Wayback Machine. This dataset could have also been downloaded directly from Archive-It. For these reasons, we consider this `CDX` generation step to be negligible in the benchmarks.

In the HBase (Warcbase) approach, we had to first ingest `WARC` files into HBase. Warcbase exploits certain properties of HBase to enable access to Web archives. For instance, different captures of a crawled Web resource are stored as timestamped versions of the same record in HBase. URLs are stored in an inverted, sort-friendly format and are used as row keys for fast lookups with the MIME type serving as a column qualifier. These design decisions allow for an efficient selection and filtering process based on these three properties: URL, timestamp of capture, and the MIME type. When additional fields are required, those need to be parsed from the `WARC` records, either from headers or the payload, which are stored as values in HBase cells. Due to limitations on the local disk space of our cluster, we had to ingest the data into HBase from the `WARC` files stored in HDFS. As the current version of Warcbase only supports reading in `WARC` files from the local file system, we modified this system accordingly. The ingesting process took a little over 24 hours with the resulting database containing a complete copy of the entire collection.

For both the Spark and HBase approaches, we queried the data using Spark and also used it to perform operations on the resulting data. All three systems being compared ran with the same Spark configurations, using 10 executors with 4 GB of memory each. As the cluster was not exclusively available to us, with other jobs running at the same time, the cluster load varied among the benchmarks. To compensate for these variations, we ran every single benchmark a total of five times.

We chose a common task among all benchmarks: select a subset of records from the entire dataset, count the length of the string content of these records and

[20] http://www.cloudera.com
[21] https://github.com/internetarchive

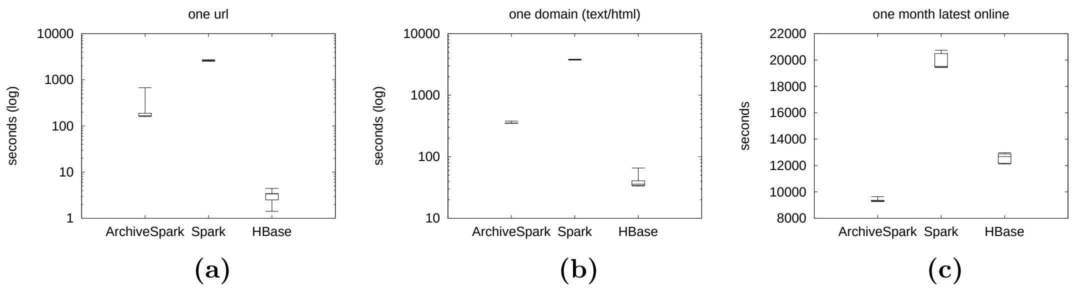


Figure 3.13. Benchmark Times of ArchiveSpark vs. Spark vs. HBase (both by leveraging Warcbase)

compute the sum of these lengths. This task is well-suited for the benchmarking process since it features the extraction workflow supported by ArchiveSpark. It involves a filtering phase to select the subset of records of interest, an enrichment phase to augment records with content, as well as a derivation phase that enriches the content with its string representation and length. We intentionally did not apply any more sophisticated enrichments that involved third-party libraries as those would only be applied on top of these results and would depend on the performance of these external tools.

Scenarios and Results

The benchmark consisted of three different scenarios, starting with the most basic filtering operation to only select records of a given URL, and ending with a more sophisticated scenario involving a grouping operation to select the latest online capture of all URL from a specific time period.

Scenario 1. First, we filtered the dataset for all records of one particular URL, i.e., *http://map.15october.net/reports/view/590/*. In case of HBase, this is directly supported and constitutes a simple row query. Therefore, it is understandably very fast with the query taking between 1.4 and 4.4 seconds. However, when comparing with the other approaches, the pre-processing time required for HBase as well as the additional space requirements need to be kept in mind (see *Experimental Setup* above). The times of all three approaches are illustrated in Figure 3.13a, where the whiskers represent the fastest and slowest runs, while the box covers the ones in the middle, with a centered line representing the median. As shown, ArchiveSpark is about 100 times slower with times between 160.3 and 675.4 seconds, but still around 10 times faster than pure Spark with times between 2522.6 and 2734.0 seconds. This is where ArchiveSpark's incorporation of the CDX index leads to performance benefits as it allows for the selective access of only records of the given URL, while pure Spark performs a scan over the entire dataset and parses every single record in order to find these records.

Scenario 2. In the second scenario, instead of filtering by URL, we selected all webpages, i.e., MIME type *text/html*, belonging to a specific domain, i.e., *15october.net*. The results are shown in Figure 3.13b. The HBase query performs a targeted row scan again, this time for all keys starting with the specified domain in its inverted, sort-friendly form, i.e., *net.15october*). However, this alone is not sufficient as the scan would also yield rows starting with *net.15octoberx*, which is not the correct domain. Therefore, an additional filtering step is required. Next, the filter by MIME type *text/html* is also directly supported by HBase, since MIME type is available as a column label. With times between 33.4 and 65.6 seconds, the HBase approach is around a magnitude of 10 slower than in the first scenario. ArchiveSpark comes closer to HBase with times between 349.2 and 379.1 seconds, because both values to be filtered are part of the `CDX` and therefore, the task is similar to the one in the first scenario. The pure Spark approach of a complete scan is around 10 times slower than ArchiveSpark with times between 3737.7 and 3853.2 seconds.

Scenario 3. Finally, we selected the latest successful captures for all URLs crawled in a specific month, i.e., Dec 2011. This is accomplished in two steps: first, all captures from the desired time period (Dec 2011) and with a successful response (status code 200) are selected and next, the latest capture for each candidate URL is chosen. The pure Spark approach takes between 19432.0 and 20744.3 seconds in this scenario. This approach first scans through all records of the dataset, followed by the step of identifying the latest capture of every URL from the set of qualifying records. This may be more efficient when only a few records of a dataset need to be filtered out. However, in scenarios, like this, where users are interested in only a small subset of a large collection, it is very slow. In the HBase approach, although HBase directly supports timestamp-based filtering, which is performed on the versions of a URL, filtering on the HTTP status code requires parsing the `WARC` record to read in the status code. Only then can the latest successful captures be selected as an additional post-processing step. The HBase approach takes between 12,117.7 and 12,971.5 seconds. For ArchiveSpark, as both properties, timestamp and HTTP status, are contained in the `CDX` files, the filtering as well as selection of the latest captures is entirely possible using just the `CDX`. For that reason, ArchiveSpark leads in this benchmark as illustrated in Figure 3.13c with times between 9639.6 and 9270.8 seconds. This illustrates how the rich potential of ArchiveSpark's selective access approach is unlocked when a large fraction of the dataset can be filtered out based on available metadata.

3.2.6 Beyond Web Archives

Books, journals, and other traditional print media items are being made available outside of physical libraries at a rapidly increasing pace. With the falling cost of high-quality digitization technologies, organizations such as *Google* and the *Open Content Alliance* are scanning books and other physical media on a massive scale and digitized content providers, such as *Internet Archive*, *The Digital*

Public Library of America, *HathiTrust*, and *Europeana*, are enabling access to rare books, manuscripts, and other special collection materials otherwise available only at geographically sparse locations. Open access publishing and *Creative Commons* licensing are moving scholarly output in front of the paywall.

Given the **sheer volume and variety** of many of these collections, they may very well be considered *Big Data*, but are still widely neglected in this field. While digitized records can be read or viewed individually, their Big Data nature allows for completely new and interesting ways of access and analysis. Instead of *close reading* every single record, the digital collection can be filtered down by specific features, enriched, and aggregated for useful statistics in a **distant reading** manner [117] (see Sec. 3.2.1).

Similar to Web archives, all the above-mentioned data sources have an important trait in common: they are maintained by libraries and archives, where records are **commonly organized in metadata indexes**. As explained before, ArchiveSpark leverages such metadata records as lightweight data proxies to provide an easy and efficient way to process *Web archives*. Subsequently, third-party tools can be easily integrated with ArchiveSpark to extract and/or derived new information to be used in *distant reading* workflows that are described in a rather declarative manner.

These workflows are not exclusive to the research of Web archives though. They can, in fact, be applied to any digital collection. Examples include digitized books and journals, which may be stored in the form of files in a specific data format or made available through a public Web service and are therefore ready for being analyzed in a *distant-reading* manner with ArchiveSpark. With its ability to integrate remote sources (see Sec. 3.2.4), ArchiveSpark can make use of any query service, e.g., databases and search engines, that enables the retrieval of metadata with pointers to the corresponding data records. Furthermore, we sought interoperability to reuse existing components of ArchiveSpark like tools that have been bundled as modules to be used with Web archives. For instance, a sentiment analysis tool that has been prepared for use with ArchiveSpark for Web archives should be usable with any text, regardless of whether it is part of a Web archive or a book corpus or any other collection. By providing this flexibility, ArchiveSpark greatly facilitates sharing of code among researchers, even across different disciplines.

With that in mind, the approach as shown in Figure 3.11 can be extended as illustrated in Figure 3.14, generalizing ArchiveSpark's two-step data access mechanism through *metadata proxies*. Together with the Medical Heritage Library (MHL), we have created a reference implementation to showcase a scenario for the work with ArchiveSpark beyond Web archives and evaluated the effectiveness of sharing job definitions [10]. MHL is a digital curation collaborative effort among some of the world's leading medical libraries. It promotes free and open access to quality historical resources in medicine. The goal of the MHL is to provide the means by which readers and scholars across a multitude of disciplines can examine the interrelated nature of medicine and society, both to inform contemporary medicine and strengthen our understanding of the world. The MHL's growing collection of digitized medical rare books, pamphlets, journals, and films number in

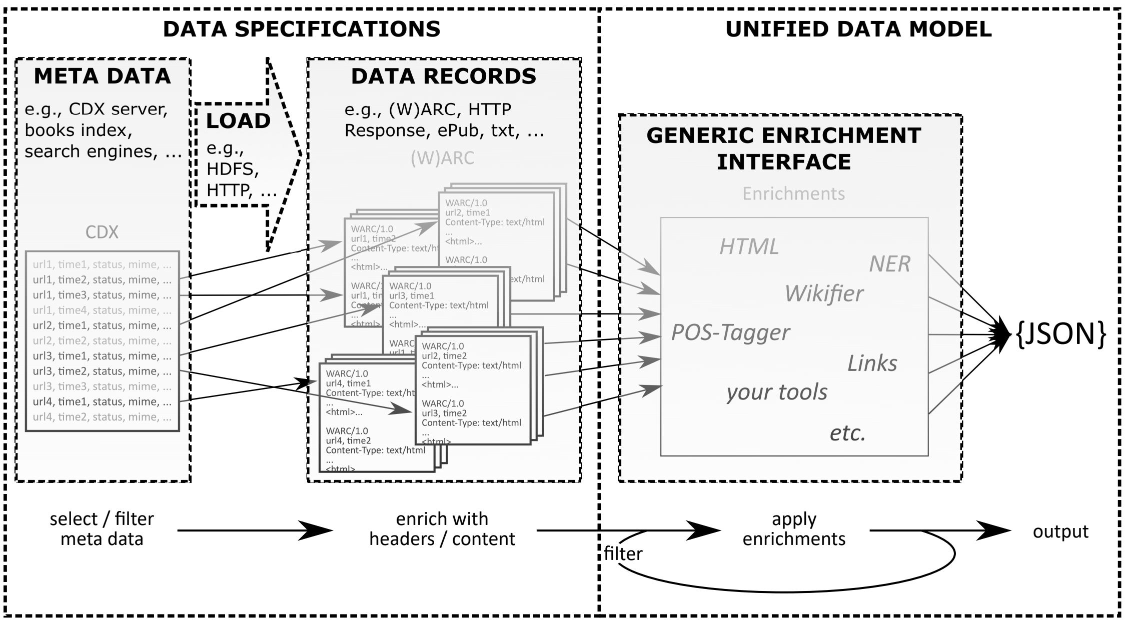


Figure 3.14. ArchiveSpark's selection and enrichment approach with variable metadata and data sources. Dashed boxes illustrate the generalizations of the original approach for Web archives.

the tens of thousands, with representative works from each of the past six centuries, all of which are available through the Internet Archive and discoverable through an advanced full-text search interface[22]. The DataSpec for querying this search system to efficiently access and study their holdings is available open-source[23].

3.2.7 Conclusion and Outlook

Web archives are becoming more and more important as a scholarly source and building a corpus from these archives is typically one of the first steps in any research process. Since researchers working with these Web collections are often from the humanities with no technical background, there is clearly a need to simplify this extraction and derivation process. Therefore, we presented a number of objectives and discussed why we deem them as essential for any system that supports building research corpora from Web archives. These include simplicity in terms of usage and extensibility, efficiency of access and traceability by documenting data lineage for the purposes of reproduction and reuse.

With ArchiveSpark, we present a framework that effectively tackles these objectives by making use of existing file formats, a functional approach to data processing at scale and utilizing a widely deployed metadata index. By utilizing this index that is a de-facto standard in the area of Web archiving, ArchiveSpark avoids having to perform any pre-processing of the data or having to invest in additional storage space. We also provide benchmarks that show how ArchiveSpark is more efficient

[22]http://mhl.countway.harvard.edu/search
[23]https://github.com/helgeho/MHLonArchiveSpark

than other alternatives when selecting records of interest based on the rich metadata already available in the metadata index. ArchiveSpark, however, is not the best option when a data processing task needs to run across all or a large fraction of the records in a Web archive.

Due to its modular architecture, ArchiveSpark serves as a universal tool for data analysis and distant reading across different data types and sources, even beyond Web archives. Its support for remote data sources, which can be streamed over HTTP or other protocols, allow researchers to efficiently extract corpora from publicly available, archives without needing a local copy of the complete dataset. By facilitating the reusability of modules and tools, workflows written with ArchiveSpark can be easily shared, even across disciplines and datasets.

To get users started more easily, we provide recipes for different analysis tasks that can be easily customized for individual needs. Further, we constantly develop new modules as well as extensions to ArchiveSpark to grow its acceptance and showcase new use case scenarios. Available add-ons include a server instance that enables to apply enrichments on a dataset through a Web service API as well as a library to create a semantic layer consisting of RDF triples from archival collections [11, 12]. The latter is presented in Chapter 4, the *graph-centric view*, as its output connects resources through knowledge graphs. In the future, we plan to grow the ecosystem around ArchiveSpark, not only by providing extensions but also specialized indexes and services that can be employed for more efficient access based on features like topics or entities contained in a document. ArchiveSpark is fully open source, and we hope for many contributions from the broader community, especially third-party tools to be provided with corresponding enrich functions to be used in the ArchiveSpark pipeline.

4 Graph-centric View: Exploring Web Archives Through Graphs

In the previous chapters, we have presented two views on the use of Web archives: from a *user's perspective*, which is focused on single documents, as well as from a more comprehensive *data-centric* point of view, in which Web archives are considered big data collections. The remaining view is from a structural perspective, focusing on the *graphs* spanning a Web archive. In the practical work with Web archives, all threes views are often combined and complement each other. They can be considered different zoom levels of an archive as introduced in Chapter 1. The *graph-centric* view constitutes the broadest zoom in this model, which allows looking at archived collections as a whole. This synoptic view allows to take the relations among the contained documents into account. Most obvious when working with the Web are relations in the form of hyperlinks that connect webpages through clickable texts or images. Just like these links are used to navigate the live Web, the graph-centric view is key to navigate in a Web archive and find the resources of interest.

In Chapter 2, we have shown with Tempas v2 how hyperlinks and their anchor texts can be exploited for searching Web archives. As pointed out, without going into details, the underlying model of that approach is, in fact, a graph. In that particular case, we considered the number of pages linking to some target page in a given time period as an indicator for the importance or popularity of the target in this period, something that would be impossible to assess without zooming out from an individual page level and taking their surroundings into account. The pages constitute the nodes of a graph in this case, with the links being directed edges among these nodes. Details and alternative ways to interpret links and construct such a graph will be introduced in Section 4.1. Furthermore, we will look at a semantic layer as a different type of graph, created on top of Web archive data, before we end with an example study that integrates all three presented views in a data analysis task, starting from the graph-centric perspective.

Finally, we want to allude to an open issue: the incompleteness of Web archives. This inherent and unavoidable characteristic of these valuable collections should always be kept in mind when working with them but has not been sufficiently studied yet. The representation of a Web archive as graph makes this especially apparent, as destinations of links may not be archived yet and hence, lead to an inconsistent state in the graph. Algorithms running on these graphs can be affected by that, resulting in deviations of the outputs compared to the same algorithm running on the full Web graph. In Section 4.2 we discuss this issue by means of the PageRank algorithm and compare rankings of incomplete graphs imposed by its results.

4.1 Web Archives as Graphs

A Web archive is a collection of pages $p \in \mathcal{P}$, each with one or multiple versions / captures $p = \{c_{p,t_1} \dots c_{p,t_n}\}$ representing the page at different points in time $(t_1 \dots t_n)$, that is when the page was crawled and archived. Each capture c_t represents a response of the Web server returned to the crawler at time t for a requested page p. This can be a successful response including the contents of the page or it could be a different status reply hinting at the page being offline or moved. In the first case, hyperlinks or other information may explicitly or implicitly reveal connections to other pages, while the latter case would likely result in a dead end in most graph representation.

The most common type of Web graphs, also for Web archives, is a hyperlink graph, with each capture modeled as set of contained links for a successful response or the empty set otherwise: A capture $c_i \subseteq L$ of page p_i is a set of links where each link $l_{ijm} \in L$ points from p_i to another page p_j, anchored by a text segment $a_m \in A$ (on p_i), i.e., the text that can be clicked on, typically called *anchor text*. In essence each link l_{ijm} connects a source page p_i with the anchor text a_m to a destination page p_j.

One of the challenges when working with Web archives, also from the graph-centric perspective, is their temporal dimension: Since the **Web is dynamic** and pages change over time, **links can change** as well. Those link changes are constituted by either:

- a link is removed from a page,
- a new link is added to a page,
- the anchor text of a link changes,
- the target page of a link changes.

A change of the source would be considered the link to be removed from its original page and added to another one. As we identify a link by its properties, i.e., an edge consisting of a source and a target page as well as its anchor text, the result of any of the four types of changes applied to a link would be considered a different link. However, a link l_{ijm} being removed from page p_i and added back later to it with the same target page p_j and the exact same anchor text a_m is treated as the same link again. Although those changes happen all the time, even multiple times between two snapshots in an archive, we can only consider those changes than are captured.

Besides Web graphs based on hyperlinks, we will look at other kinds of graph, and focus on semantic knowledge graphs in particular. In contrast to hyperlink graphs, these connect resources based on semantic relations, like those pages or captures that mention the same person. For the creation of any of these graphs, the holdings of a Web archive need to be processed in order to extract relations. With ArchiveSpark (see Sec. 3.2) we presented a tool to do this efficiently. Now, we will show an extension of ArchiveSpark to extract facts in the form of RDF triples and construct a graph, called semantic layer, for Web archives. A semantic layer constitutes another form of graph on top of a Web archive, which helps to get an overview and identify connections among the contained documents.

Eventually, we outline an experiment in which we demonstrate how all three presented views can be integrated in a data processing workflow to scale up to thousands of pages. Starting from the graph-centric view by integrating Tempas (see Ch. 2), we investigate exemplarily the increase of restaurant prices in Germany when the Euro as a new currency was introduced, as reported by an offline study [123].

4.1.1 Related Work

Works specifically on graphs in Web archives are very limited, but scientists have looked into graph properties of the Web in general both on static [69, 70, 71, 72, 73] and evolving graph [74, 75, 76]. However, it is worth looking at graphs in Web archives as a special case of Web graphs, because archives are never complete and not crawled with a particular attention to preserving the original Web graph structures. At the same time though, they are our only source to analyze the Web of the past and its evolution retrospectively. As a consequence, questions on the completeness and consistency of graphs extracted from Web archives arise, such as: *How well do structures and properties of graphs extracted from a Web archive resemble the graph of the actual Web and what is the impact of missing pages on the behavior or results of graph algorithm, like* PAGERANK*?* [68]. These aspects have often been neglected in the past and need to be studied in more detail in the future. Initial results on the issue on incompleteness of Web archives will be touched upon by us in Section 4.2.

A specific type of graphs that researchers have investigated by means Web archive data, is social networks. With SocGraph, Shaltev et al. [48] present an exploration system for social relations among people and their temporal evolution, extracted from the content of a Web archive. They describe methods for constructing large social graphs from extracted relations and introduce an interface to study their temporal evolution. While in their case, the graph is purely constructed of people and relations, this semantic information could be connected to resources in the archive where it is extracted from, described by available metadata, and in this way provide a semantic layer based on the RDF/S data model on top of the raw archive [124]. In contrast to other user-centric solutions for exploring Web archives (cf. Ch. 2, such a semantic layer could satisfy more complex information needs by exploiting the expressive power of SPARQL [125] and its federated features [126, 11]. By connecting resources through semantic knowledge, like social data, it provides a graph-centric view on the archive. Further, semantic technologies allow to connect it with existing information available on the Linked Open Data cloud [127], like DBpedia [128]. In that way, we are able not only to explore archived documents in a more advanced way, but also integrate information, infer new knowledge and quickly identify interesting parts of a Web archive for further analysis.

A similar approach has been recently proposed by Page et al. [129]. In this work, the authors build a *layered* digital library based on content from the Internet Archive's live music collection. Starting from the recorded audio and basic information in the archive, this approach first deploys a *metadata layer* which allows an initial consolidation of performer, song, and venue information. A *processing layer* extracts audio features from the original recordings, workflow provenance, and summary feature metadata, while a *further analysis* layer provides tools for the user to combine audio and feature data, discovered and reconciled using interlinked catalogue and feature metadata from the other layers. Similar to our approach, the resulting layered digital library allows exploratory search across and within its layers. However, it is focused on music digital libraries and requires the availability of a large amount of metadata which is not usually the case in Web archives. On the contrary, our approach focuses on *entity-centric* analysis and exploration of an archived collection of documents.

Similar to Tempas for exploring a Web archive powered by an underlying hyperlink graph, user-friendly interfaces can be developed on top of semantic layers to facilitate their use for end-users as well. Systems like Sparklis [130] and SemFacet [131] already enable the exploration of semantic repositories through a faceted search interface [132, 133]. Other approaches translate free-text queries to SPARQL [134]. Providing such interfaces on top of graphs, like hyperlink graphs or semantic linked data graphs, connects the broad graph-centric view with the user-centric view on Web archives, which highlights the synergies among them.

4.1.2 Hyperlink Graph Models

Following the notion introduced as introduced above, we now define different types of graphs in a Web archive, based on hyperlinks. Let $G = (V, E)$ be a graph with a set of nodes / vertices $v_i \in V$ and a set of directed edges $e_{jk} = (v_j, v_k) \in E \subseteq V \times V$ pointing from node v_j to v_k. For the Web or a Web archive, such a graph represents the links connecting webpages: each node $v_i \in V$ represents a webpage and an edge $e_{jk} \in E$ corresponds to a link from webpage v_j to page v_k. Two or more links on the same page with the same destination and anchor text are considered to be equal and only counted as one. However, it is possible that multiple links $l_{jkm} \in L$ exist for the same edge e_{jk}. Hence, the Web $\mathcal{W}$ can be modeled as an edge-labeled, directed graph, with the labels being the set of anchor texts accompanied by a set of links: $\mathcal{W} = (G, A, L)$.

In Web archives, which have a temporal dimension, there are multiple ways to **extract a representation of the Web** W as defined above **for a time interval** $[t_a, t_b]$. As the graph G as well as the set of anchor texts A in $\mathcal{W}$ can be derived from the set of links L, the temporal representation depends on how we extract the links from the captures during the given interval: $L \subseteq \bigcup_{p \in \mathcal{P}} c \in \{c_t \in p | t \geq t_a \wedge t \leq t_b\}$. While there exist various ways to define this set, we identified the following three:

1. **Merge**. The most straight-forward way is to merge all links that existed in some capture during the time interval under consideration. Hence, the resulting graph also includes links that existed in some capture but were deleted later in another capture of the same page during the interval. Two or more links corresponding to the same edge but labeled with different anchor texts at different time points during the interval are all included:

$$L_{\texttt{merge}} = \bigcup_{p \in \mathcal{P}} c \in \{c_t \in p | t_a \leq t \leq t_b\}$$

This is the most complete representation as it merges all links, regardless of whether they still exist at the end of the interval or not.

2. **Temporal Snapshot**. A temporal snapshot or simply *snapshot* refers to a single point in time as opposed to a period. For a given time interval $[t_a, t_b]$, the snapshot representation at time t_b contains only links from the latest captures c_t for each page from that interval with $t \leq t_b$, i.e., links that actually exist at time t_b:

$$L_{\texttt{snapshot}} = \bigcup_{p \in \mathcal{P}} \underset{c_t \in \{c_t \in p | t_a \leq t \leq t_b\}}{\arg\max} \; t$$

This model resembles the actual Web at time t_b most closely and is best suited for analyzing the Web or its structure. However, it is not ideal for information retrieval as we miss the intermediate states in the considered time period corresponding to the granularity of our index.

3. **Emergence**. *Emergence* refer to the links posted / created in a given time interval $[t_a, t_b]$. In that respect it is as complete as a *merged* representation if all consecutive intervals are considered. However, it is more space efficient as it does not contain the captures already present before the considered interval:

$$L_{\texttt{emergence}} = \bigcup_{p \in \mathcal{P}} c \in \{c_t \in p | t_a \leq t \leq t_b\} \setminus \bigcup_{p \in \mathcal{P}} c \in \{c_t \in p | t < t_a\}$$

This representation has two major advantages for temporal search, which is the reason why we chose $L_{\texttt{emergence}}$ to build our indexes for Tempas v2 (see Sec. 2.1.5): 1) only pages that are actively being linked in the queried time period are considered, while pages that got frequently linked earlier but are not relevant anymore are ignored even if the old links still exist on the Web, 2) the index size is significantly reduced as each link is only included in one interval.

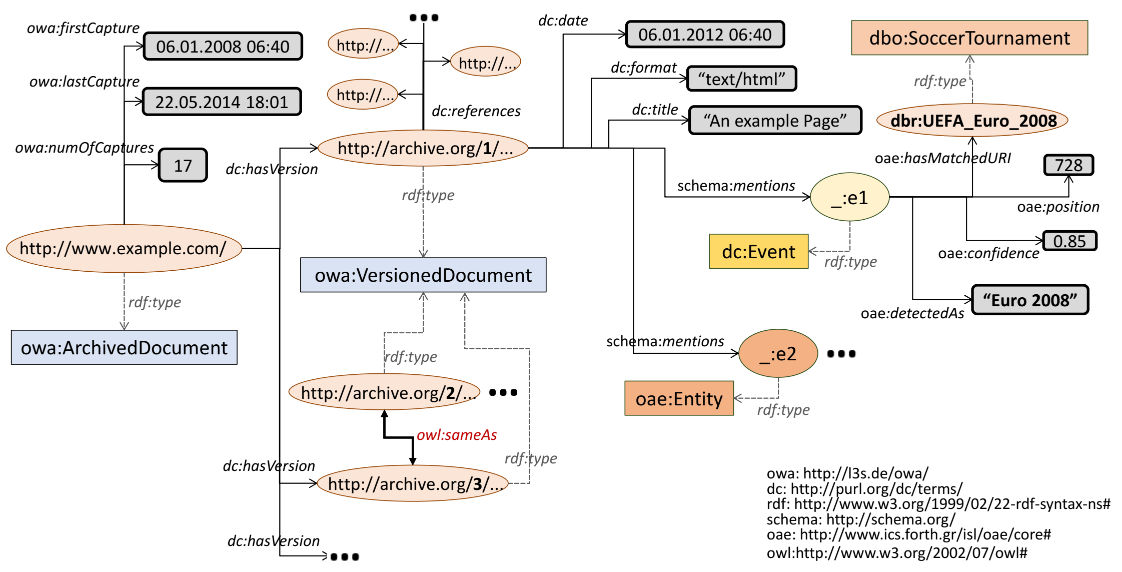


Figure 4.1. Semantic layer describing an archived webpage using the *Open Web Archive* data model [11].

4.1.3 Semantic Layers for Web Archives

As semantic layer, we conceive a graph representation of a Web archive that connects semantic information extracted from the contents of archived webpages with the metadata of the corresponding captures. An example of that is shown in Figure 4.1. Such a layer constitutes a way of describing metadata information about the archived documents together with annotations with useful semantic information, like entities, concepts and events, and publishing all this dataset on the Web as Linked Data base on Semantic Web technologies by means of RDF triples [124].

In order to investigate the effectiveness of such a semantic layer in terms of exploring an archive, we constructed the required triples for the *Occupy Movement 2011/2012* collection, which was already used to evaluate ArchiveSpark in Section 3.2.5. For each version of an archived webpage, we stored its *capture date*, its *title*, its *mime type* and its *extracted entities*, while for each distinct URL we stored its total number of captures, the date of its first capture, and the date of its last capture. As resource identifiers (URI) of the versioned webpages, we assigned the corresponding location in Archive-It's Wayback Machine.

The semantic layer contains 1,344,450 *same-as* properties, which means that we avoided annotating and storing identical information for a very large number of versioned webpages. Moreover, 939,960 distinct entities (including concepts and events) were extracted from the archived webpages. For each entity, we stored its name (surface form), its URI, its position in the text, and its confidence score. In total, the constructed semantic layer contains more than 10 billion triples (10,884,509,868).

Construction

For the construction of the semantic layer, we developed ArchiveSpark2Triples[1], an extension of ArchiveSpark (see Sec. 3.2) that outputs extracted or derived information from the resources in a Web archive in the Notation3 (N3) RDF format based on the *Open Web Archive* data model [11]. Internally, ArchiveSpark2Triples defines three types of documents: *archived document* (instance of `owa:ArchivedDocument`), *versioned document* (instance of `owa:VersionedDocument`), and *same-as versioned document* (instance of `owa:VersionedDocument`, which constitutes a *revisit-record*, i.e., duplicate of a previous capture). In more detail:

- An *archived document* represents all versions of the same webpage, i.e., all records with the same URL. Its triples reflect the webpage as one unit, including the number of captures in the Web archive, the timestamps of the first and last capture as well as pointers to the corresponding *versioned documents*.
- A *versioned document* represents each individual capture of a webpage, i.e., every record of a webpage in the archive. The assignment of URIs to the versioned documents is customizable and thus can be defined by the user. By default, the triples of such a document only include the date of the capture and its mime type (e.g., text, image, etc.). However, the framework supports the extension of this by accessing and transforming any properties stored in ArchiveSpark's data model into triples. ArchiveSpark's enrich functions are supported as well in order to derive and seamlessly integrate information from the content of a webpage, e.g., the title of a page, its links to other pages, and its entities. For the extraction of entities, we used *Yahoo's Fast Entity Linker (FEL)* [135]. The enrich function required to make it useable with ArchiveSpark is available under FEL4ArchiveSpark[2].
- A *same-as versioned document* represents an already archived webpage that has not changed from the previous capture. In this case, only a *same-as* property pointing to the earlier record is created. The exact way in which duplicates are identified is not part of the framework and can be defined as part of the generation workflow. We use the digest/hash value that is part of the `CDX` records for this, which allows to do it in an efficient way and avoid unnecessary accesses to the archived content of the duplicates.

Finally, the vocabularies used by the produced triples can be defined as part of the generation workflow and thus can be customized by the user as well.

[1]`https://github.com/helgeho/ArchiveSpark2Triples`
[2]`https://github.com/helgeho/FEL4ArchiveSpark`

Efficiency

ArchiveSpark2Triples gains its efficiency from the efficiency of ArchiveSpark, which is mainly a result of the two-step approach that is used for data loading and access [9]. To recap Section 3.2, an archived collection to be used with ArchiveSpark consists of two parts, the `WARC` records containing the actual data with headers and payloads, and the `CDX` records containing only basic metadata, such as URLs, timestamps and datatypes, which are considerably smaller in size. Hence, operations that rely exclusively on information contained in the metadata can be performed very efficiently, e.g., filtering out items of a certain type. Eventually, if operations need to be performed on the actual contents, only the required records are accessed using location pointers in the `CDX` records. ArchiveSpark2Triples benefits from this approach, since records of a datatype other than *text/html*, such as images and videos, can be filtered out very fast. In addition, all properties of the *archived documents* and the majority of properties of the *versioned documents* can be generated purely based on metadata and thus, very efficiently. In fact, the payload is accessed only for applying *enrich functions*, e.g., for extracting the title of a webpage, its entities, which is only done for the *versioned documents* without duplicates, as described above.

The most expensive task in our pipeline is the entity extraction process, performed by FEL4ArchiveSpark using Yahoo FEL [135]. To avoid extraordinarily long runtimes, FEL4ArchiveSpark supports to define a timeout, which we set to 10 seconds per record in our experiments. Additionally, we considered only webpages with a compressed size of less than 100 KB, as larger file sizes are unlikely to constitute a webpage and may indicate a malformed record. Although the described steps are considered quite efficient, the actual time for the entire workflow depends on the dataset size, the nature of the data as well as the used computing infrastructure. The Hadoop cluster used in our experiments for producing the semantic layer consisted of 25 compute nodes with a total of 268 CPU cores and 2,688 GB RAM. While the available resources strongly depend on the load of the cluster and vary, we worked with 110 executors in parallel most of the time, which resulted in a runtime of 24 hours for processing the entire collection of 474.6 GB of compressed `WARC` and `CDX` files.

The execution time of a SPARQL query over a semantic layer mainly depends on the following factors:

- The efficiency of the triple store hosting the semantic layer (e.g., in-memory triple stores are more efficient).
- The efficiency of the server hosting the triple store (available main memory, etc.).
- The query itself since some SPARQL operators are costly (like `FILTER` and `OPTIONAL`). Moreover, if the query contains one or more `SERVICE` operators (like the queries in Listing-4.2), then its execution time is also affected by the efficiency of the remote endpoints at the time of the request.

Effectiveness

```
SELECT ?journ (COUNT(DISTINCT ?page) AS ?num) WHERE {
SERVICE <http://dbpedia.org/sparql> {
       ?journ a yago:Journalist110224578 }
       ?page a owa:ArchivedDocument ;
       dc:hasVersion ?version .
       ?version schema:mentions ?entity .
       ?entity oae:hasMatchedURI  ?journ .
} GROUP BY ?journ ORDER BY DESC(?num)
```

Figure 4.2. SPARQL query for retrieving the most discussed journalists in webpages of the *Occupy Movement* collection.

By querying a semantic layer, we can infer useful knowledge related to the archived documents that is very laborious to derive otherwise. The Occupy Movement collection that our semantic layer has been constructed for, contains webpages related to protests and demonstrations around the world calling for social and economic equality. A reasonable query in this context would be for the most discussed journalists mentioned on the webpages of the collection, as shown in the listing in Figure 4.2. Notice that the query counts the archived documents, not the versions, in order to avoid counting the same page captured in different time periods multiple times. The query returns *Ralph Nader*, *Chris Hedges* and *Dylan Ratigan*, as three of the most discussed journalists. Obtaining the same information from a pure keyword-based search, such as Tempas, would not be as straight-forward, since keywords alone are not suitable to formulating the considered information. Also, the system would only be able to return a set of candidate pages, which a user would need to read in detail in order to find the requested information. The possibility of formulating the complex queries in expressive SPARQL makes a semantic layer well-suited for questions like the above.

Further, due to the graph nature of linked data in a semantic layer, which connects the contained information with corresponding documents of origin, the approach also facilitates the navigation in an archive. On the other hand, without a more user-friendly interface, querying and using a semantic layer is not appropriate for end users like addressed in the user-centric view in Chapter 2. However, in combination with a data processing tool like ArchiveSpark, it could serve for identifying entry points in the data-centric view (Ch. 3) for downstream data analysis tasks, similar to the experiment presented in the next section.

4.1.4 Data Analysis by Incorporating Graphs

Finding the right entry points into a Web archive is crucial, not only for manual exploration (cf. Ch. 2), but also for data analysis tasks. Due to the vast sizes of Web archives in the order of hundreds of terabytes or even petabytes, scanning all pages is impossible. In this respect, the graph-centric perspective on the archived data as exploited by our temporal search system Tempas, can help to find the entry points over time and reduce the dataset of relevance for the temporal data analysis task. From the identified pages or captures, the archive can be scanned further. Although this does not guarantee a full coverage of all interesting contents, it is likely to detect relevant pages and thus, constituting an efficient way to create a representative sample supporting significant analysis results. As an example, we

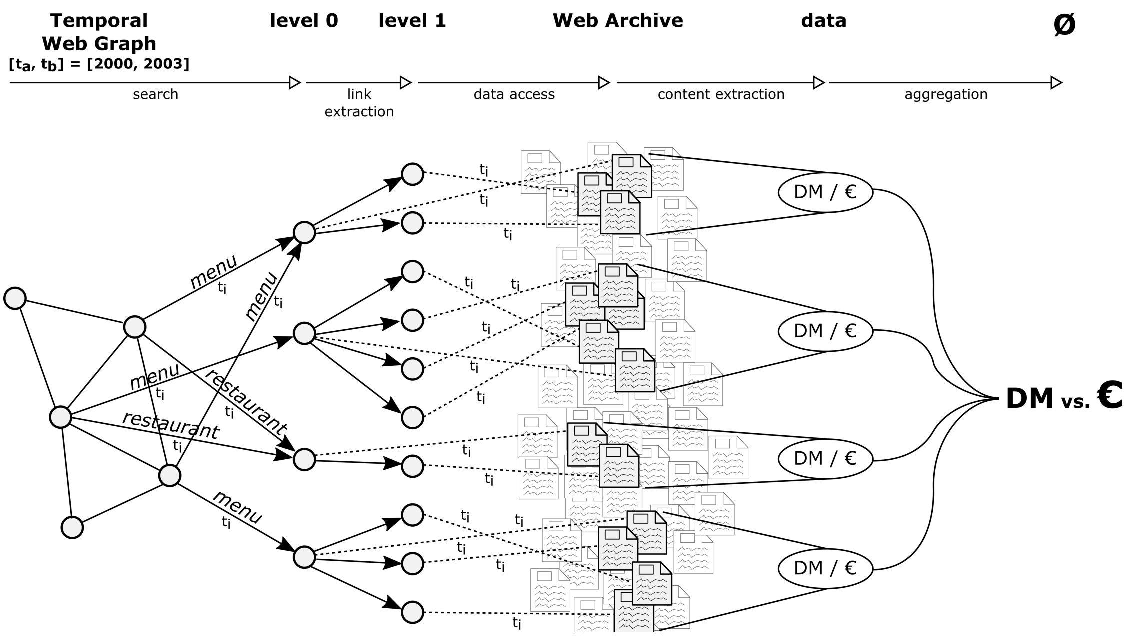


Figure 4.3. Data analysis pipeline, starting from Web archive search based on a temporal hyperlink graph with corresponding anchor texts, over selective Web archive data access and extraction, to the aggregated results presented to the user.

studied the evolution of restaurant prices in Germany during the time when the Euro was introduced as Europe's new currency.

Setup

Figure 4.3 outlines the steps performed in this experiment by integrating the three views on Web archives, similar to the generic framework for systematic data analysis presented in Chapter 1 (Fig. 1.2). Starting from the graph-centric perspective to identify suitable entry points, we transition to the data-centric analysis, before results are aggregated and prepared to be presented to the user. In addition to that, the user-centric view allows inspections at all stages of the pipeline.

Towards this, we integrate Tempas v2 (see Sec. 2.1) as an alternative metadata provider for ArchiveSpark (see Sec. 3.2), while the archived webpages corresponding to the search results are loaded from the Wayback Machine. Hence, a pre-filtering of the Internet Archive's Web archive is performed by keyword as well as time through the Tempas index, based on the underlying hyperlink emergence graph as described above (Sec. 4.1.2), before content is loaded for returned hits.

The logic for this data loading and integration step is defined in a separate data specification module for ArchiveSpark, named Tempas2ArchiveSpark. It has been published on GitHub[3] together with the code that describes the pipeline for this particular analysis as illustrated in Figure 4.3. Since metadata and data

[3]https://github.com/helgeho/Tempas2ArchiveSpark

records are loaded remotely, the study can be repeated or similar studies can be conducted by anyone even without having a suitable dataset available.

Example Study

In this small example study we utilize ArchiveSpark to analyze menus of German restaurants at the time when the Euro was introduced as the new currency in Europe and replaced the former German currency *Deutsche Mark (DM)* in 2001/2002. According to many voices in Germany this resulted in increased prices particularly in restaurants. In 2011 the German federal office of statistics published a report in which they studied the effect of the Euro in various areas and categories [123]. According to that study, restaurant prices increased around the time of currency reform by about 4% on average based on more than 700 examined restaurants. However, while some restaurants even reduced their prices, others increased them by up to 20% to 40%.

To get entry points into the archive for our analysis, we queried Tempas for the keywords '*speisekarte*', which is the German word for menu, as well as '*restaurant*' to find menus after one additional hop, in case it is not linked with the term for menu in the anchor text. Both queries were issued for the time period from 2000 to 2003 and we considered the first 10 pages with 100 results per page, hence 1,000 hits each. As some of the returned URLs were found in multiple years, we started off with 3,567 hits of URL / timestamp pairs for which we integrated contents from the Wayback Machine By manually investigating a few of the result pages on Tempas, we found that they often do not show the prices directly, but link to subpages, such as for starters or main courses. Therefore, we extracted all links under the same hostname and fetched the contents for those URLs as well, which resulted in additional 6,028 pages. From each of these pages we extracted the amounts of money mentioned on the pages as floating-point numbers prefixed or suffixed with either *DM*, *Euro*, *EUR* or the Euro symbol €. To verify that we were actually dealing with restaurant menus, we required the pages to mention the term *speise*, which is the German word for meal / dish, and to contain at least three prices. This filtering left us with menus for 49 entry point URLs.

After converting DM to Euro amounts with an exchange rate of 1 EUR = 1.95583 DM, we averaged the amount per currency and URL as well as among URLs. Finally, we ended up with an average of 9.58 Euros and 15.24 DM = 7.79 Euros. That is an increase of 23%. Although this is higher than the average 4% as reported by the study mentioned above, it is in the range between 20% and 40% that was found in the statistics report. We can therefore consider it a reasonable result, although not significant due to the limited number of analyzed menus. The experiment shows, however, how Web archives can be used for studying social issues with temporal anchor texts serving as entry points to access and analyze these extremely big collection, which are otherwise almost impossible to process. Thus, given a better domain knowledge of the studied subject and by investing some additional effort, we should be able to produce interesting and significant results with this approach.

4.1.5 Conclusion and Outlook

Graphs provide a synoptic view on data. Already in Chapter 2 we have shown how hyperlinks serve as effective cues for identifying temporally relevant documents in a Web archive. Graph representations are the key to such methods by connecting the otherwise individual captures in a Web archive. A labeled hyperlink graphs as the most obvious and basic concept already goes a long way but also raises open challenges. One of them is caused by the incompleteness in Web archives that we will touch upon in Section 4.2.

Based on an emergence graph, as one of three presented types of hyperlink graph models for Web archives, that powers our Tempas search engine, we have outlined an example study of restaurant menus to demonstrate the possibilities of integrating graphs in data analysis tasks. With 49 menus out of the initially 3,567 hits and 6,083 analyzed URLs, studying Web archives often is like finding a needle in a haystack, though. This stresses the need of effective ways to identify entry points for both manual exploration or *close reading* as well as data analysis or *distant reading* tasks.

As an alternative type of graph, we believe that semantic layers are a first step towards more advanced and meaningful exploration of Web archives. However, for future work, user-friendly interfaces should be developed on top of semantic layers for allowing end-users to easily and efficiently explore Web archives. Another interesting direction is to study approaches for ranking the results returned by SPARQL queries [136, 137].

4.2 On the Incompleteness of Web Archives

Most real-world graphs collected from the Web like hyperlink graphs and social networks are *incomplete* or in other words their graph topology is not known in entirety [138, 139]. Especially if not crawled for a particular purpose or subset, but extracted from existing crawls, such as Web archives. The goal of Web archive crawlers is to capture as much as possible starting from some seed set within some national domain or even broader, given the available but limited resources [140]. Incompleteness is an inherent trade-off already in the design decision of such an archive [141]. Complicating matters further, Web archives are often not constructed in one piece but by merging partial crawls [8]. Additional reasons for the incompleteness in Web archives include the restrictive *politeness* policies (i.e., *robots.txt*) or random timeouts of Web servers. Several studies on this topic have shown that incompleteness is indeed a common issue [96, 142, 143, 144], inevitably affecting the graphs extracted from such crawls as well.

As a result, important graph properties and measures used for link analysis and structural characterization like *authority of vertices* might be inherently flawed or exhibit deviations from their original values. This is commonly observed where users are typically agnostic to the incompleteness of the obtained graph, hoping that the input graph is a reasonable representative sample of the underlying (unseen) original graph. Some well-known measures for computing authority of vertices or relative ordering of vertex authorities based on random walks are PageRank [68] and its variants [145, 146, 147].

As an example, consider PageRank computed over the `.gov` Web graph that we will analyze in detail later in this work. Here, the `women.nasa.gov` (*Women@NASA*) page has a high PageRank value and is subsequently found within the top 300 pages. However, on a closer examination we observe that most of its PageRank is contributed by an in-link from the highly popular NASA homepage (`nasa.gov`). If for some reason this particular in-link is not crawled, e.g., due to a temporary downtime or the decision by *NASA* to exclude their homepage from being crawled, this would cause a large decrease in its PageRank and hence a severe rank deviation in the obtained crawl.

While one might argue that this is an unlikely case, since *important* pages enjoy a high priority and are therefore commonly crawled, this is not the case in reality. We found, by ranking pages in a graph constructed from a `.de` Web archive in 2012, first, based on their in-links and second, based on their PageRanks, that indeed, even prominent pages are often missed. The graph considered only links that emerged in 2012 (cf. Sec. 4.1.2) [6], but we compared against all pages archived in that year. According to in-links only roughly 30% and according to PageRank less than 20% of the top 1000 pages are not contained in the archive. The latter one is *questionable* though, as our results will show, the ranking imposed by PageRank on an incomplete graph is way less stable than we have commonly thought.

Therefore, we study the deviation in orderings/rankings imposed by PageRank over incomplete graphs. Vertices in our input crawls are either *completely crawled* (all neighbors are known) or are *uncrawled* (none of their neighbors are known), which we refer to as *ghost vertices*. Based on this, the research questions we ask are the following:

- **RQ I :** *Do incomplete real-world graphs show a deviation in their* PageRank *orderings when compared to full network topology?*
- **RQ II :** *How can we reliably measure the extent of such ranking deviations for incomplete graphs?*

Towards these, we perform extensive experiments on both real-world Web and social network graphs with more than 100 million vertices and 10 billion edges. We first establish empirically that real-world networks indeed show a deviation in their PageRank orderings when not crawled completely compared to the complete graph (**RQ I**). We observe ranking correlations (measured by *Kendall's Tau*) dropping down to 0.55 on Web graphs when only 50% of it is crawled. Second, users and applications that use rankings induced by PageRank as a feature for downstream ranking and learning tasks would naturally be interested in estimating such a deviation from the (incomplete) input graph at hand as a measure of confidence. Therefore, as an answer to **RQ II**, we propose a measure called HAK that estimates the ranking deviation of an incomplete input graph when compared to the original graph [13, 14].

4.2.1 Related Work

Ng et al. [148] analyzed the conditions under which eigenvector methods like PageRank and HITS can provide reliable rankings under perturbations to the linkage patterns for a given collection. In particular for PageRank they showed that, if perturbed or modified webpages, i.e., links from the page are removed or are not followed, did not have a high PageRank score in the original graph, then the new PageRank score will not be far from the original. However, similar to our results, they also sometimes observed strong deviations in the rankings imposed by the algorithms when more important pages are perturbed. In their paper, they discuss these only for the top 10 items in either of the considered rankings though. We on the other hand, provide a quantitative evaluation using Kendall's Tau for a much larger fraction of the graph, which is crucial for the use of PageRank in Information Retrieval scenarios where only a selected set of relevant documents are ranked. Further, we provide a measure to estimate ranking deviations of vertices in the given graph with respect to their orderings in the original unmodified graph.

Boldi et al. [149] also show the paradoxical effects of PageRank computation on Web graphs. In contrast to our work though, they discuss a measure of effectiveness for crawl strategies based on whether the graph obtained after a partial visit is in some sense representative of the underlying Web graph for the PageRank computation. Similar to our setting, they study how rapidly the computation of PageRank over the visited subgraph yields relative ranks, measured by Kendall's Tau, that agree with the ones the vertices have in the complete graph.

In [150], unlike other approaches that sample vertices, the authors operate on a given subset of vertices and consider the general problem of maintaining multi-scale graph structures by preserving a distance metric based on PageRank among all pairs of sampled vertices.

The other area of related work comprises graph sampling approaches which can be broadly classified into two categories: *traversal based* methods [151, 152, 153] and random walk based methods [138, 139, 154]. Graph-traversal based methods employ breadth-first search (BFS) or the depth-first search (DFS) algorithm to sample vertices and are typically shown to exhibit bias towards high-degree vertices [152]. Maiya and Berger-Wolf [153] compare various traversal based algorithms and define representativeness of a sample while proposing how to guide the sampling process towards inclusion of desired properties. On the other hand, the random walk based methods are popular for graph sampling because they can produce unbiased samples or generate samples with a known bias [155, 138, 139, 154]. Other related works deal with estimating graph properties such as degree distribution estimation [138], clustering coefficient estimation [156], size estimation [157], and average degree estimation [158]. However, most of these works assume a known graph topology. Our work focuses on the unknown graph topology, an arguably more general and useful scenario in Web graphs and social networks gathered by crawlers.

4.2.2 Preliminaries and Problem

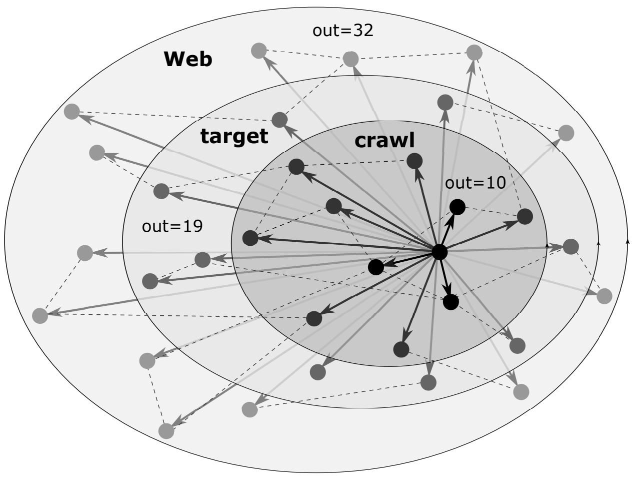


Figure 4.4. The neighborhood of a webpage in different subgraphs of the Web: The out-degree differs as neighbors become ghost vertices in the target graph or crawl. While the target represents the desired subset to be crawled, the crawl illustrates what has actually been captured, making this an incomplete graph.

As originally conceived, PageRank ranks vertices of a directed graph $\mathcal{G} = (V, E)$ where V and E are the vertices and edges respectively, based on the topological structure of the graph using random walks [68]. The problem we are addressing is attributed to this random walk model behind PageRank, representing the *authority* or *importance* of a vertex. For some fixed probability α, a surfer at vertex $v \in V$ jumps to a random vertex with probability α and goes to a linked vertex with probability $1 - \alpha$. The *authority* of a vertex v is the expected sum of the *importance* of all the vertices u that link to v. Consequently, a vertex receives a high PageRank value and is ranked at the top by ordering the webpages by *importance* when it is either connected by many incoming edges or reachable from another *important* page.

We first define the notions of *target graph*, *crawl* and *ghost vertices* in the context of incompleteness in graphs due to their collection process:

Definition 1 (Target graph)**.** *The subset of vertices (with the induced edges) of a larger graph (e.g., the Web) that is theoretically reachable by a crawler given its seeds, e.g., a domain, a top-level domain, or all webpages that belong to a certain topic in case of focused crawlers. This graph would be available if every link was followed and every page captured by the crawler, illustrated by the* target *in Figure 4.4.*

Definition 2 (Crawled graph or Crawl)**.** *The (incomplete) graph derived from the set of webpages that have actually been visited by the crawler, discovered/linked yet uncrawled pages are not included. This subset of the target graph is illustrated by the* crawl *in Figure 4.4.*

Definition 3 (Ghost vertex)**.** *Although a hyperlink on a crawled page points to another page that belongs to the target graph, there is a chance the crawler never visited and saved that page, i.e., it is not part the crawl. Such a page or vertex is referred to as* ghost vertex, *shown by the gray vertices outside the crawl in Figure 4.4.*

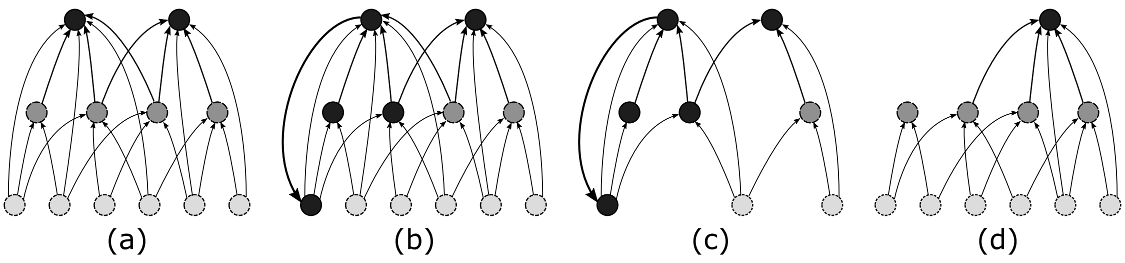


Figure 4.5. Some graph structures: A darker color of the vertices represents a higher *importance*.

The deviation among two rankings induced by PageRank is a global objective, independent of a specific query. Hence, local or relevance-based measures such as nDCG are not applicable here. The most common metrics to quantify rank correlation are *Spearman's Rho* and *Kendall's Tau*, which are both similar as they are special cases of a more general correlation coefficient and measure relative displacements. In this work, we use Kendall's Tau [159], ranging from $[-1, 1]$, with 1 corresponding to a perfect rank correlation, 0 corresponding to no correlation and -1 to a perfect inverse correlation. This is used to compare the correlation/deviation of rankings computed on the vertices of a crawl $\mathcal{G}_C$ with respect to that of the target graph $\mathcal{G}_T$.

In Figure 4.5 we provide a few examples of possible graph structures, where partial knowledge of the graph may affect the ranking returned by the PageRank values. We remark that in the next sections, we will also provide empirical evidence, supporting the fact that there exist ranking deviations in crawls of some real-world graphs. In the first subfigure (a), we show the positive case of a DAG where the partial knowledge of the graph will not cause any ranking deviations. As only the topmost vertices shown here receive significantly more links than the others, these are also the most *important* vertices. It is easy to see here that generating a crawl from this structure by removing some vertices will not cause any significant changes in the ranking orderings of the crawl. In the next subfigure (b), a *backlink* has been introduced (left) that feeds back the importance of a top most page to a previously unimportant page and its successors. This importance gets propagated through the cycle which has been created due to the inserted *backlink*. In the next subfigure (c), we illustrate the case of a crawl in which vertices are removed uniformly at random. The chances here are that primarily unimportant vertices are removed, which would still not cause severe deviations in the ranking orderings. Finally, if we remove any vertex from the cycle as shown in subfigure (d), its succeeding vertices drastically lose in importance and hence, the ranking among the pages in the crawl changes noticeably.

4.2.3 The HAK Measure

The goal of this measure is to estimate quantitatively how reliable a crawl is with respect to the relative ordering of the PageRank values on its vertices compared to the corresponding target graph. To this end, we first try to **estimate the size of the target graph**: Given the crawled vertex set and the distinct hyperlinks on the corresponding webpages, some of which are pointing to an uncrawled page (ghost vertex), how big is the target graph or a subgraph that would potentially impact or contribute to the PageRank values of the vertices in the crawl? We show that for simple crawling strategies where it can be assumed that each vertex is part of the crawl independently from all other vertices with some sampling probability p_s, the size of the target graph can be estimated in terms of a very simple property of the crawled vertices, namely, the fraction of its crawled neighbors, referred to as *fidelity*. Secondly, we try to **estimate the impact** exerted by the vertices in the target graph on the crawled vertices, which we in turn use to estimate the number of discordant pairs in the expected rankings, like in Kendall's Tau.

Let C denote the set of vertices of the *crawl graph* and let n be the number of vertices in this graph. The main steps in our computation are as follows:

1. Estimate the size of the target graph by using connectivity properties of the crawl. Let T represent the set of vertices in this target graph.
2. Estimate the *impact* (as functions of PageRank) of the vertices in C.
3. Assume that the vertices in T exert similar impacts on other vertices.
4. Estimate the number of discordant pairs due to impacts exerted by vertices in $T - C$ on vertices in C.

Estimating the Target Graph

Let $\mathcal{N}$ denote the number of vertices in the target graph. In this section we will estimate the value of $\mathcal{N}$ under the simplified assumption that the crawl is constructed by sampling vertices from the target graph independently and uniformly at random with some probability p_s. Note that if p_s is known, one can easily estimate $\mathcal{N}$ as $\frac{n}{p_s}$. We therefore first estimate p_s from the connectivity of the crawl, using a property that we refer to as **fidelity**: For any vertex $v \in T$, we define fidelity $(\gamma(v))$ of v as the ratio of its immediate neighbors in C to its total out-degree (number of distinct hyperlinks on a webpage pointing to vertex in T). Let $d_c(v)$ count the number of vertices $v' \in C$ reachable from v in one step. $d(v)$ denotes the total out-degree of v in the target graph. This results in the following definition:

Definition 4 (Fidelity)**.** *The fidelity of a vertex $v \in T$, $\gamma(v)$, is given by $\gamma(v) = \frac{d_c(v)}{d(v)}$ and the average fidelity of all vertices in C is*

$$\gamma(C) = \frac{\sum_{v \in C} \gamma(v)}{n}$$

With p_s as the sampling probability, $p_s \cdot \mathcal{N}$ would be the number of vertices in the crawl. Hence, using the observed average $\gamma(C)$ and the observed size of the crawl (n), we approximate $\mathcal{N}$ as $\frac{n}{\gamma(C)}$.

PageRank and Impacts

Despite its incompleteness, PageRank can be computed on the crawl graph by treating the ghost nodes as dangling nodes. We use the *personalized* variant of PageRank for this, starting from the available nodes in C as seeds (see Sec. 4.2.4). Given this, for any vertex v in the crawl C, let $\pi(v)$ denote the value computed by PageRank and let $N(v)$ denote the set of succeeding neighbors of v, reachable from v in one step, hence $d(v) = |N(v)|$. PageRank of any vertex u can now be considered as:

$$\pi(u) = \sum_{v:u\in N(v)} \frac{\pi(v)}{d(v)}$$

Based on these considerations, we introduce a new property, referred to as **impact**. The impact of a vertex $v \in C$ on one of its neighbors $u \in N(v)$ is defined as:

$$Im(v,u) = \frac{\pi(v)/d(v)}{\pi(u)}$$

Hence, the total impact on any vertex $u \in V$, received from all its incoming edges, is $\frac{1}{\pi(u)} \sum_{v:u\in N(v)} \frac{\pi(v)}{d(v)}$, which is always 1. This implies that any extra impact of x on a vertex will increase its PageRank by x times the current PageRank.

The total impact of a vertex v, $Im(v)$ is then defined as:

$$Im(v) = \sum_{u\in N(v)} Im(v,u) = \sum_{u\in N(v)} \frac{\pi(v)/d(v)}{\pi(u)} = \frac{1}{d(v)} \sum_{u\in N(v)} \frac{\pi(v)}{\pi(u)}.$$

We denote the average of impacts of vertices in C by $Im(C)$, i.e.,

$$Im(C) = \frac{\sum_{v\in C} Im(v)}{n}$$

Estimating the Impact of Ghost Vertices

We next compute the impact that could have been exerted by the ghost vertices on the crawled vertices, if the graph was complete and the ghost vertices existed. In a setting like ours, where the (*personalized*) PageRank is computed from the perspective of the known crawl (see above), the ghost nodes cannot have a bigger impact on the crawl than previously *leaked* to them. Therefore, we build on the assumption that the impact of each vertex in the complete target graph T is on average the same as for the crawl: $Im(C)$. Hence, we approximate the impact exerted by ghost vertices only as follows:

$$\mathcal{I} = |T - C| \cdot Im(C) = n\left(\frac{1}{\gamma(C)} - 1\right) \cdot Im(C).$$

Some of this extra impact, generated due to ghost vertices, will be acquired by some or all of the vertices in C, changing their PageRank values accordingly. This is what eventually will lead to the deviation in rankings, measured by Kendall's Tau as the number of pairs of each two crawled vertices $(v, u) \in C \times C$ for which the order differs, i.e., *discordant pairs*, or is preserved, i.e., *concordant pairs*. Since HAK is meant to predict the deviation as assessed by Kendall's Tau, we also estimate both these classes of pairs in order to compute HAK.

The impact of the ghost vertices can be divided among the vertices of the crawl in several ways. For example, it can happen that the vertex with the lowest PageRank receives the total impact, increasing its PageRank by a large factor. In this case the number of discordant pairs is upper bounded by $n-1$. Moreover, we know from [148] that vertices with low original PageRank scores will also have a low PageRank value in slightly modified graphs. Therefore, the effect of the loss of information because of incomplete crawls is observed mostly on the PageRanks of the nodes higher in the original ranking. We checked experimentally several variants for impact distributions and the best variant, which is affirmative with our tests on real-world graphs, is to distribute the total impact $\mathcal{I}$ equally among $\mathcal{I}$ vertices. Hence, the **expected number of impacted vertices** that belong to the crawled set will be:

$$I = \mathcal{I} \cdot \gamma(C).$$

In the worst case, each of these impacted vertices will result in forming a discordant pair with each of the unaffected vertex, resulting in a number of discordant pairs of $D = (n - I) \cdot I$. Based on that, HAK is computed with respect to Kendall's Tau as follows:

$$\begin{aligned} HAK =& \frac{\#\text{concordant pairs - }\#\text{discordant pairs}}{\#\text{ total pairs}} \\ =& \frac{\frac{n(n-1)}{2} - D - D}{\frac{n(n-1)}{2}} = 1 - 4 \cdot \frac{D}{n(n-1)}. \end{aligned}$$

4.2.4 Experimental Setup

For the experiments we require the availability of crawls as well as the complete target graphs that these crawls were derived from. This is necessary in order to compute how the rankings on both graphs differ and to evaluate the performance of HAK to estimate this deviation. In reality, however, neither obtaining the complete target graph is possible nor the actual crawl policy can be determined accurately. To this extent, we employ very large real-world graphs that themselves were incomplete and considered them as target graphs by discarding edges that connect to ghost vertices. With these graphs, we simulated crawls in order to generate new incomplete subgraphs. For all these crawls we ran PageRank on both graphs (crawl and target graphs) and compared the rankings using Kendall's Tau to evaluate HAK.

The experiments were run on a computer cluster using *Apache Spark* and its graph processing framework *GraphX* [160]. Loading the graphs locally on a single

	GOV	**DE**	**UK**	**Friendster**
$\#V$	301,128,778	247,641,473	39,454,746	68,349,466
$\#V_{target}$	5,418,054	133,895,590	38,838,959	61,100,375
$\#E$	2,111,229,433	14,795,732,782	936,364,282	2,586,147,869
$\#E_{target}$	180,657,788	10,085,242,536	928,939,162	2,575,600,737

Table 4.1. Statistics on the Studied Graphs ($\#V$: original number of vertices, $\#E$: original number of edges, $\#V_{\text{target}}$: target number of vertices, $\#E_{\text{target}}$: target number of edges)

server was impossible with our available infrastructure because of their sizes of up to more than 100M vertices and 10B edges, summarized in Table 4.1:

- **GOV** : This graph is based on crawled webpages provided by the Internet Archive. It was extracted from the latest captures of all their archived webpages under the `.gov` top-level domain (TLD) from 2005 to 2013.
- **DE** : Like GOV, this `.de` TLD graph was also extracted from webpages archived by the Internet Archive, crawled in 2012.
- **UK** : This `.uk` TLD crawl from 2005 is publicly available, already in the form of a graph without corresponding webpages [73, 161].
- **Friendster** : Unlike the previous Web graphs, this is a publicly available social network, extracted from an extensive crawl of the former online platform *Friendster.com* in June 2011 [162].

Additional experiments to generalize our approach and study its utility for a wider range of different graph topologies by means of synthetic graphs are presented in Holzmann et al. [13, 14]. These reveal the effect of incompleteness with respect to specific properties and characteristics that are also partially reflected by the studied real-world Web graphs.

Seed Selection and Crawling

Crawling can be considered a special case of network sampling from a more practical point of view, where subsequent vertices can only be chosen from already discovered ones or seeds. Maiya and Berger-Wolf [153] define this type of sampling as *link-trace sampling* and give a nice overview of available models for this behavior. Naturally, such approaches commonly exhibit BFS-like (Breadth-First Search) growth but feature different strategies to prioritize or select the next vertices to be crawled. These variations determine the probability of a vertex to be part of the final sample.

Although most crawlers employ BFS-like traversals, there are practical constraints like random timeouts and crawl restrictions on websites that make it hard to model crawls perfectly. Therefore, we focus on the most impartial strategy, which is vanilla BFS, but explicitly **produce partial crawls** by dropping $x\%$ of the vertices of the input graph (where $x \in \{10, 20, 30, 40, 50\}$). We refer to this percentage as the *block fraction* and the remainder as *desired fraction*.

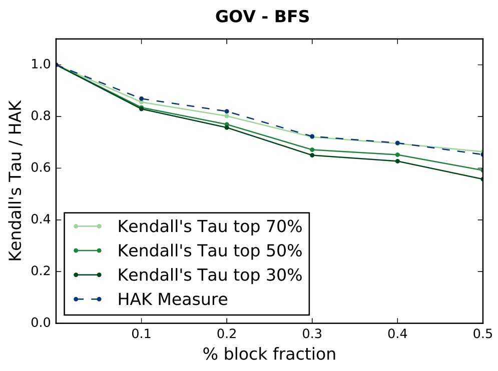
Statistics about the *target graphs* (V_{target} and E_{target}), which are potentially reachable from the seeds by not blocking any vertices are shown in Table 4.1. The breadth-first search (BFS) starts from a set of seed vertices and runs until all vertices are reached. A number of vertices according to the block fraction were chosen uniformly at random and blocked/discarded before the BFS, simulating vertices that cannot be crawled, e.g., due to *robots.txt*, slow response times, etc.

We found out that the most realistic seed selection strategy is to pick the most important vertices as seeds. This is also the case for real crawls as these correspond to more well-known pages. To identify such pages in our target graphs, we first ran PageRank on them and constructed the seed set from the top 1%. This allowed us to reduce the size of the large real-world graphs by pre-computing the actual target graphs, consisting only of vertices that are reachable from the seeds (see Table 4.1, V_{target} and E_{target}). Interestingly, for the GOV and DE graphs, the size difference of the target graphs compared to the originally provided graphs is huge, which confirms common characteristics of these Web archive graphs, i.e., they are not constructed in one crawl, leading to a fairly large number of unimportant vertices (with no in-edges) that were discovered from crawls outside target graphs. The UK and Friendster graphs on the other hand remained at almost the same size, suggesting that they have already been created that way in the first place, which proves our seed selection strategy actually realistic.

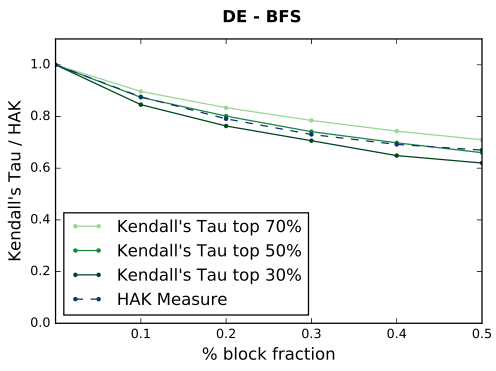
Evaluation strategy

The objective of this evaluation is to assess ranking deviations as quantified by Kendall's Tau (cf. Sec. 4.2.2) for rankings induced by PageRank, computed on a complete target graph vs. an incomplete crawl and compare it against our HAK measure, which is designed to yield values on the same scale. For this, we **focus only on high-ranked vertices**, as these are typically more interesting in most practical scenarios [148]: Firstly, because there is no tangible score difference between the PageRank values of the tail vertices. Secondly, ranking deviations in authoritative vertices are typically considered more severe than among the tail ones. Since Kendall's Tau makes no distinctions where rank reversals take place, we compared the ordering among the top 30%, top 50% and top 70% vertices of the crawl and target graph that appeared in both graphs according to the corresponding PageRank values. This also helps us characterize where the rank reversals indeed do appear.

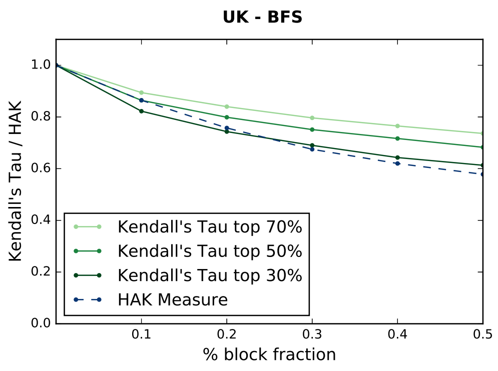
The rankings for each of the graphs are computed based on the PageRank values. While we employed the regular version PageRank on the crawl (with added ghost vertices as sinks), we used the *personalized* variant of PageRank for running it on the target graph. In this version, the algorithm is personalized to a set of vertices, which constitute the starting points as well as teleportation destinations in the algorithm [68]. The resulting PageRank values can be interpreted



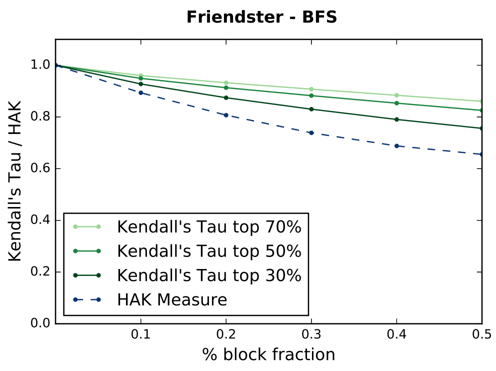
(a)



(b)



(c)



(d)

Figure 4.6. Ranking deviations measured and estimated for real-world graphs and crawls for different fractions of uncrawled vertices.

as their importance with respect to these vertices or the domain represented by the crawl. Both variants of PageRank ran for 30 iterations with the damping factor parameter set to the frequently cited value of 0.85.

4.2.5 Observations and Results

In the following we describe our observations on ranking deviations caused by the incompleteness of the underlying graphs or the Web archives datasets that these graphs have been extracted from, respectively. Further, we assess the effectiveness of our HAK measure in order to evaluate our assumptions on implications of incompleteness as described in Sections 4.2.2 and 4.2.3. The computed deviations with respect to Kendall's Tau as well as estimated values based on HAK for the crawls on all four graphs are presented in Figure 4.6.

We first focus on **RQ I** and justify the need for estimating ranking deviations before employing PageRank for incomplete graphs. We clearly observe noticeable ranking deviations of partial crawls with respect to target graphs as all four graphs exhibit a decreasing Kendall's Tau with increasing block fraction. Most significantly is the drop to 0.55 for the GOV. However, the trends are very similar for all studied graphs. The least severe deviations or smallest decrease of Kendall's Tau is exhibited by the Friendster graph, our only social network graph, which does not reflect a natural Web structure. We believe this is because the scale-free nature of social graphs is much stronger than of Web graphs, and hence, they are less affected by missing nodes [163]. We further observe that the ranking deviations increase when we consider a smaller fraction of the most important vertices. This indicates that most of the low rank vertices in the target graph do not flip their ranks with the more important ones in the crawl, leading to a lower ratio of discordant pairs to the overall total number of pairs. Hence, popular nodes or pages tend to remain rather popular, even in incomplete graphs, however, the relative order among then varies a lot.

We now turn to **RQ II** and evaluate whether the assumptions we have constructed our HAK measure on actually hold. We recall that one of the main assumptions behind HAK is that each of the unseen or ghost vertices from the target graph would exert the same fraction of impact (on average) to the crawled set as the actual vertices in the crawl (cf. Sec. 4.2.3). We ensure this by constructing the target graph such that each of its vertex has the same fraction of crawled neighbors as the crawled vertices (computed by fidelity). This assumption would not be followed by target graphs, if the ghost vertices had no edges back into the crawl (backlinks).

However, our results support the effectiveness of HAK in the studied graphs and therefore also validate our assumptions behind HAK. For instance, for the UK graph we report an almost precise estimation (actual: 0.58, estimated: 0.61). In contrast, the deviation in Friendster is less strong and slightly overestimated by HAK (actual: 0.76, estimated: 0.66), which might be due to its different nature as pointed out above and the consequent scarcity of *backlinks*. Overall, all the results are promising though. We also notice that our estimates reflect more closely the ranking deviations among the top PageRank vertices, which we believe to be more interesting for most practical purposes than deviations in less *important* vertices or the entire graph.

4.2.6 Conclusion and Open Challenges

We studied the problem of incompleteness in Web archives and the consequences on derived graphs, leading to deviations in rankings imposed by graphs algorithms, such as PageRank. We showed that these in fact do occur and can be drastic, as shown in our GOV graph where the correlation among the rankings among the complete graph and a crawl is only 0.55, measured by Kendall's Tau. To this effect, we proposed the HAK measure to estimate such deviations purely on the crawl without any knowledge of the original graph, based on assumptions about the impact of incompleteness in Web archives on graphs.

The fact that the results of PageRank can actually be discorded due to incomplete graphs should be considered when employing PageRank to order crawled webpages, as it can lead to severe ranking deviations between the incomplete crawl and the original target graph. The proposed HAK measure is an attempt to estimate this deviation purely on the crawl without any knowledge of the original graph, which turns out to be remarkably reliable.

In future work, we would like to generalize our approach further for a wide range of different graph topologies. Towards this, we have been conducting more comprehensive studies by means of synthetic graphs, which allow us to systematically analyze the effect of incompleteness with respect to specific properties or characteristics of the underlying dataset [13, 14]. Also, we would like to investigate the applicability of our measure to determine the confidence of results produced by other algorithms on incomplete graphs, such as random walk algorithms similar to PageRank.

5 Conclusion and Future Work

Web archives have been instrumental in digital preservation of the Web and provide great opportunities for the study of the societal past and evolution. These archival collections are massive datasets, typically in the order of terabytes or petabytes, spanning time periods of up to more than two decades and growing. Due to this, their use has been difficult as effective and efficient exploration as well as access methods are limited. We have identified three views on Web archives, for which we propose novel concepts and tools to tackle existing challenges: user-, data- and graph-centric.

The natural way to look at a Web archive is through a Web browser, just like regular users explore the live Web as well. This is what we consider the *user-centric view*. The most common method to access a Web archives from a user's perspective is the Wayback Machine, the Internet Archive's replay tool to render archived webpages. Those pages are identified by their URL and a timestamp, referring to a particular version of the page. In order to facilitate the discovery of a page if the URL is unknown, we proposed different approaches to search Web archives by keywords through external, temporal cues [2, 4, 6]. Another way for users to find and access archived pages is by linking information from the past on the current Web to the corresponding evidence in a Web archive [3, 5, 7].

Besides accessing a Web archive by closely reading pages in a browser, contents can be analyzed through distant reading, too. In this *data-centric view*, webpages are not considered self-contained units with a layout and embedded assets. Instead, individual resources are processed and analyzed as raw data like text or images [8]. These data analysis tasks usually do not involve the whole archive, but only a certain time period, specific data types, or other facets that can be filtered on first. By taking this into consideration, we proposed a novel approach for building research corpora from Web archives by incorporating lightweight metadata records to process Web archives at scale in a very efficient manner [9, 10].

The third perspective, besides the user-centric and data-centric views, is what we refer to as the *graph-centric view*. Instead of individual resources or pages, the focus here is on the relations in Web archives. These are, in the most basic case, hyperlinks among the archived pages [6]. By regarding contents and semantic information, more expressive graphs can be constructed to enable complex queries and exploration patterns [11, 12]. This structural perspective allows completely different kinds of analysis and exploration by forming synergies with both other views. At the same time it exposes issues inherent in Web archives, such as their incompleteness, that must not be neglected in practical use [13, 14].

5.1 Scientific Contributions

Towards tackling the raised challenged from above, we have presented the following scientific contributions with respect to each of the three addressed views on the use of Web archives:

- User-centric View: Browsing the Web of the Past.

- We identify that for typical access methods to Web archives, which are navigational and temporal in nature, we do not require indexing full-text. Instead, meaningful text surrogates like anchor texts already go a long way in providing meaningful solutions and can act as reasonable entry points to exploring Web archives. By taking this into consideration, we present **new approaches to searching Web archives** based on surrogates, such as tags attached to social bookmarks, as well as temporal link graphs and corresponding anchor texts. Departing from traditional informational needs, we show how these can be effective in answering queries beyond purely navigational intents, like finding the most central webpages of an entity in a given time period. We propose indexing methods and a temporal retrieval model based on anchor texts, as well as demonstrate and discuss several interesting search results using our approach, as implemented by our Tempas (Temporal archive search) system.

- We show that most meaningful information about entities or objects, such as software, like descriptions, metadata, documentation and source code, is usually available online. Especially for these complex and evolving entities, traditionally referenced metadata schemas are often not expressive enough to capture their temporal states comprehensively. We show how Web archives can be helpful in this respect, by serving as **rich digital object representation to be linked from the live Web or cited in literature**. Currently though, we found that only 10% of the studied blog posts and roughly 30% of the analyzed software websites are archived completely, i.e., all linked resources are captured as well. Therefore, we propose Micro Archives to ensure a coherent archived state among logically connected resources. With Micrawler we present a modular solution to create, cite and analyze such Micro Archives. We show the need for this approach and discuss opportunities as well as implications for various applications.

- Data-centric View: Analyzing Archival Collections.

- The Web has been around and maturing for about 25 years. The popular websites of today have undergone vast changes during this period, with a few being there almost since the beginning and many new ones becoming popular over the years. Therefore, we conducted a **longitudinal study, spanning almost the whole period of the Web**, based on data collected by the Internet Archive starting in 1996, to retrospectively analyze how the popular Web as of now has evolved over the past 18 years. For our study, we focused on the German Web, specifically on the top 100 most popular websites in 17 categories, and present a selection of the most interesting findings in terms of *volume*, *size* as well as *age* of the Web. We found that around 70% of the pages we investigated are younger than a year, with an observed exponential growth in age as well as in size up to now. In addition to that, we give detailed insights into our methodology, **purely based on lightweight, easily shareable metadata**, to foster the replication of similar studies in the future.

- The analytical use of Web archives at scale requires tools that provides efficient access to the holdings for data scientists and researcher. We identified five requirements based on practical needs, such as ease of use, extensibility and reusability. To meet these objectives, we propose ArchiveSpark, **a flexible framework for efficient, distributed Web archive processing** based on existing and standardized data formats commonly held by Web archiving institutions. Performance optimizations in ArchiveSpark, facilitated by the use of a widely available metadata index format, result in significant speed-ups compared to existing approaches, without depending on any additional data stores. At the same time, usability is improved by seamlessly integrating filters and derivations with external tools, even for the work with archival collections beyond Web archives.

- Graph-centric View: Exploring Web Archives Through Graphs.

- Graphs provide a synoptic view on Web archives and surface relations among otherwise disconnected resources by taking structural information like hyperlinks into account. The benefits of this for search were already considered before and shown with Tempas. We now discussed different models to extract hyperlink graphs with respect to the dynamically changing Web. Further, semantic layers were introduced as alternative graph model that offers advanced query capabilities and allows for the integration with other knowledge bases. Finally, we demonstrate the integration of the **graph-centric view to navigate in archives and identify entry points** as part of a data analysis pipeline.

- Most real-world graphs collected from the Web or extracted from a Web archive like hyperlink graphs and social network graphs are incomplete. We draw the attention to this issue by investigating its effect on graph algorithms. We first show that **incompleteness actually has an impact on graph algorithms** like PageRank and measure how much a ranking induced by these could deviate from the original unseen graph. Furthermore, we present an attempt to approximate those rank deviations in the absence of the complete graph. Our experiments on real-world graphs with more than 100M vertices and 10B edges showcase the impact of incompleteness in Web archive graphs as well as the potential effectiveness of a prediction measure for this effect.

5.2 Software Contributions

As part of the work presented in the previous chapters, the following tools have been developed as proposed solutions to the addressed problems or to support the conducted research:

Tempas. The temporal Web archive search engine Tempas exists in two versions: v1 is based on a social bookmarks dataset, v2 is based on hyperlinks and corresponding anchor texts extracted from the German Web archive. In addition to a textual query, it allows to select a time period to search in. Tempas is presented in detail in Section 2.1 and can be accessed on `http://tempas.L3S.de`.

TimePortal. The TimePortal was developed as viewer to display the results of Tempas v2. It has later been reused to contextualize an archived website linked from an object reference on the live Web. This is demonstrated for software mentioned in scientific publications by the integration in `http://swmath.org`, described in Section 2.2.

Micrawler. This framework can be used to create rich digital object representations, called Micro Archives, as introduced in Section 2.2. It can be customized to work with different Web archives and other services. Created Micro Archives, consisting of a coherent set of captures that represent a common object or entity can be shared and cited. Micrawler is fully open source and available on `https://github.com/helgeho/Micrawler`, with a reference implementation deployed under `http://tempas.L3S.de/Micrawler`.

ArchiveSpark. A general-purpose framework for efficient data processing for Web archives and archival collections based on Apache Spark, developed in cooperation with the Internet Archive. Details on the design and architecture of ArchiveSpark are described and evaluated in Section 3.2. It is freely available on `https://github.com/helgeho/ArchiveSpark` with a comprehensive documentation as well as multiple recipes to get common tasks around Web archive processing done more easily. Additionally, we provide several extensions and modules for ArchiveSpark, among others:

- **ArchiveSpark-server.** A server application that provides a Web service API for ArchiveSpark to be used by third-party applications to integrate temporal Web archive data with a flexible, easy-to-use interface: https://github.com/helgeho/ArchiveSpark-server
- **ArchiveSpark2Triples.** This library provides tools to convert ArchiveSpark records from Web archives to RDF triples in Notation3 (N3) format. It was used to create the semantic layer in Section 4.1.3: https://github.com/helgeho/ArchiveSpark2Triples
- **Tempas2ArchiveSpark.** A data specification for ArchiveSpark to load results from Tempas and integrate corresponding captures seamlessly from the Wayback Machine, like demonstrated by the data analysis experiment in Section 4.1.4: https://github.com/helgeho/Tempas2ArchiveSpark
- **MHLonArchiveSpark.** This data specification for ArchiveSpark provides the required components to work with journals from the Medical Heritage Library (MHL), like addressed in Section 3.2.6: https://github.com/helgeho/MHLonArchiveSpark

Miscellaneous. In addition to the above listed tools, many other open-source projects were created to provide different useful utilities for the work with Web archives. The following selection and more are available on GitHub https://github.com/helgeho:

- **Web2Warc.** An easy-to-use and highly customizable crawler that enables the creation of little Web archives in `WARC` / `CDX` as used by ArchiveSpark. It was used to acquire the data for our experiments in Section 2.2: https://github.com/helgeho/Web2Warc
- **HadoopConcatGz.** A *splitable* Hadoop input format for concatenated GZIP files. With this format, compressed `WARC` files can be loaded more efficiently. It is also used by ArchiveSpark to load `WARC` files without corresponding `CDX` records: https://github.com/helgeho/HadoopConcatGz
- **HadoopWebGraph.** A Hadoop input format for graphs in the BVGraph[1] format, which is widely used to compress large Web graphs, like the UK graph in Section 4.2 [73]: https://github.com/helgeho/HadoopWebGraph

5.3 Future Work

Besides presenting novel concepts and tools to facilitate the use of Web archives, we have also raised new challenges and paved the way for future work. A temporal search system like Tempas is essential for navigating in these huge archival collections and to find suitable entry points or the desired temporal resources. However, search based on surrogates like tags and anchor texts is not appropriate for every

[1]http://webgraph.di.unimi.it

use case scenario and all intents [45]. For instance, certain changes in a page that may be of high relevance to a user, might not be reflected by external sources. Hence, it is inevitable to look into alternative retrieval and ranking models in the future as well. Also, different visualizations and ways to explore archives are subject for further investigation [164]. Towards making Tempas a fully production-ready system, also other challenges need to be tackled in the future in order to enable the same convenience that we are used to in common search engines on the current Web [165]. For instance, query suggestions and reformulations, which support the users in expressing their information needs, are not available for archives yet.

Graphs have proven to be fundamental in exploring Web archives and the presented semantic layer is a first step towards more advanced and meaningful graph representation by incorporating contents as well as external semantic information. However, to effectively benefit from these approaches in the future, user-friendly interfaces that facilitate the use for end-users need to be developed on top. Further, the effect of traits specific to Web archives, like their inherent incompleteness, especially on graphs, should be studied in more detail and taken into account in future applications that are potentially affected, such as ranking [166, 137]. Other properties and the evolution of the Web may have effects on the acquisition and use of Web archives as well. What we have learned about its growth and size can impact resource allocation strategies. The introduced notations and definitions provide a solid foundation for comparing our findings on growth and aging against different Web archive collections like national holdings of other countries [167, 168].

Another direction of future work will be on building infrastructures and growing ecosystems around the presented software and tools. In terms of Micrawler, we will explore case-specific implementations of the integrated services to be hosted with partners, like persistence providers for assigning guaranteed permanent identifiers, such as DOIs. We also want to establish a platform for sharing crawl specifications and existing Micro Archives. In order to foster the data-centric use of Web archives, we will continue to develop ArchiveSpark and promote its use especially in scientific disciplines like Digital Humanities. The framework is fully open source, and we hope for many contributions from the community to integrate third-party tools by providing suitable modules. Besides, we also plan to grow the ecosystem around ArchiveSpark by hosting it as a service and providing specialized indexes that can be employed for an even more effective and efficient use of Web archives.

Bibliography

[1] Helge Holzmann and Thomas Risse. Accessing Web Archives from Different Perspectives with Potential Synergies. In *Researchers, Practitioners and Their Use of the Archived Web*, London, UK, jun 2017. School of Advanced Study, University of London. doi: 10.14296/resaw.0001. at the 2nd International Conference on Web Archives / Web Archiving Week (RESAW/IIPC).

[2] Helge Holzmann and Avishek Anand. Tempas: Temporal Archive Search Based on Tags. In *Proceedings of the 25th International Conference Companion on World Wide Web - WWW '16 Companion*. ACM Press, 2016. doi: 10.1145/2872518.2890555.

[3] Helge Holzmann, Mila Runnwerth, and Wolfram Sperber. Linking Mathematical Software in Web Archives. In *Mathematical Software – ICMS 2016*, pages 419–422. Springer International Publishing, 2016. doi: 10.1007/978-3-319-42432-3_52.

[4] Helge Holzmann, Wolfgang Nejdl, and Avishek Anand. On the Applicability of Delicious for Temporal Search on Web Archives. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval - SIGIR '16*, Pisa, Italy, 2016. ACM Press. doi: 10.1145/2911451.2914724.

[5] Helge Holzmann, Wolfram Sperber, and Mila Runnwerth. Archiving Software Surrogates on the Web for Future Reference. In *Research and Advanced Technology for Digital Libraries, 20th International Conference on Theory and Practice of Digital Libraries, TPDL 2016, Hannover, Germany*, Hannover, Germany, 2016. doi: 10.1007/978-3-319-43997-6_17.

[6] Helge Holzmann, Wolfgang Nejdl, and Avishek Anand. Exploring Web Archives through Temporal Anchor Texts. In *Proceedings of the 2017 ACM on Web Science Conference - WebSci '17*, Troy, New York, USA, 2017. ACM Press. doi: 10.1145/3091478.3091500.

[7] Helge Holzmann and Mila Runnwerth. Micro Archives as Rich Digital Object Representations. In *Proceedings of the 10th ACM Conference on Web Science - WebSci '18*, Amsterdam, Netherlands, 2018. ACM Press. doi: 10.1145/3201064.3201110.

[8] Helge Holzmann, Wolfgang Nejdl, and Avishek Anand. The Dawn of Today's Popular Domains - A Study of the Archived German Web Over 18 Years. In *Proceedings of the 16th ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '16*, pages 73–82, Newark, New Jersey, USA, 2016. IEEE, ACM Press. doi: 10.1145/2910896.2910901.

[9] Helge Holzmann, Vinay Goel, and Avishek Anand. Archivespark: Efficient Web Archive Access, Extraction and Derivation. In *Proceedings of the 16th*

ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '16, pages 83–92, New York, NY, USA, 2016. ACM. doi: 10.1145/2910896.2910902.

[10] Helge Holzmann, Vinay Goel, and Emily Novak Gustainis. Universal Distant Reading through Metadata Proxies with Archivespark. In *2017 IEEE International Conference on Big Data (Big Data)*, Boston, MA, USA, dec 2017. IEEE. doi: 10.1109/bigdata.2017.8257958.

[11] Pavlos Fafalios, Helge Holzmann, Vaibhav Kasturia, and Wolfgang Nejdl. Building and Querying Semantic Layers for Web Archives. In *Proceedings of the 17th ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '17*. IEEE, jun 2017. doi: 10.1109/jcdl.2017.7991555.

[12] Pavlos Fafalios, Helge Holzmann, Vaibhav Kasturia, and Wolfgang Nejdl. Building and querying semantic layers for web archives (extended version). *International Journal on Digital Libraries*, Jul 2018. doi: 10.1007/s00799-018-0251-0. URL https://doi.org/10.1007/s00799-018-0251-0.

[13] Helge Holzmann, Avishek Anand, and Megha Khosla. What the HAK? Estimating Ranking Deviations in Incomplete Graphs. In *14th International Workshop on Mining and Learning with Graphs (MLG) - Co-located with 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, London, UK, 2018.

[14] Helge Holzmann, Avishek Anand, and Megha Khosla. Delusive pagerank in incomplete graphs. In *Complex Networks and Their Applications VII*. Springer International Publishing, 2019. ISBN 978-3-030-05411-3.

[15] Nina Tahmasebi, Gerhard Gossen, Nattiya Kanhabua, Helge Holzmann, and Thomas Risse. Neer: An Unsupervised Method for Named Entity Evolution Recognition. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 2553–2568, dec 2012.

[16] Helge Holzmann, Gerhard Gossen, and Nina Tahmasebi. Fokas: Formerly Known As - a Search Engine Incorporating Named Entity Evolution. In *Proceedings of the 24th International Conference on Computational Linguistics: Demonstration Papers (COLING 2012)*, pages 215–222, dec 2012.

[17] Helge Holzmann, Nina Tahmasebi, and Thomas Risse. BlogNEER: Applying Named Entity Evolution Recognition on the Blogosphere. In *3rd International Workshop on Semantic Digital Archives (SDA) - Co-located with 17th International Conference on Theory and Practice of Digital Libraries (TPDL)*, volume 1091, pages 28–39, Valletta, Malta, 2013.

[18] Helge Holzmann and Thomas Risse. Named Entity Evolution Analysis on Wikipedia. In *Proceedings of the 2014 ACM Conference on Web Science - WebSci '14*. ACM Press, 2014. doi: 10.1145/2615569.2615639.

[19] Elena Demidova, Nicola Barbieri, Stefan Dietze, Adam Funk, Helge Holzmann, Diana Maynard, Nikolaos Papailiou, Wim Peters, Thomas Risse, and Dimitris Spiliotopoulos. Analysing and Enriching Focused Semantic Web Archives for Parliament Applications. *Future Internet*, 6(3):433–456, jul 2014. doi: 10.3390/fi6030433.

[20] Helge Holzmann and Thomas Risse. Extraction of Evolution Descriptions from the Web. In *IEEE/ACM Joint Conference on Digital Libraries*, pages 413–414, London, UK, sep 2014. IEEE Press, IEEE. doi: 10.1109/jcdl.2014.6970201.

[21] Helge Holzmann and Thomas Risse. Insights into Entity Name Evolution on Wikipedia. In *Web Information Systems Engineering – WISE 2014*, pages 47–61, Thessaloniki, Greece, oct 2014. Springer International Publishing. doi: 10.1007/978-3-319-11746-1_4.

[22] Helge Holzmann, Nina Tahmasebi, and Thomas Risse. Named Entity Evolution Recognition on the Blogosphere. *International Journal on Digital Libraries*, 15(2-4):209–235, apr 2015. doi: 10.1007/s00799-014-0135-x.

[23] Tarcisio Souza, Elena Demidova, Thomas Risse, Helge Holzmann, Gerhard Gossen, and Julian Szymanski. Semantic URL Analytics to Support Efficient Annotation of Large Scale Web Archives. In *Semantic Keyword-based Search on Structured Data Sources*, pages 153–166. Springer International Publishing, 2015. doi: 10.1007/978-3-319-27932-9_14.

[24] Anett Hoppe, Jascha Hagen, Helge Holzmann, Günter Kniesel, and Ralph Ewerth. An Analytics Tool for Exploring Scientific Software and Related Publications. In *Research and Advanced Technology for Digital Libraries, 22nd International Conference on Theory and Practice of Digital Libraries, TPDL 2018*, Porto, Portugal, 2018. doi: 10.1007/978-3-030-00066-0_27.

[25] Jefferson Bailey et al. (Internet Archive). Web Archiving in the United States: A 2013 Survey, 2014. URL http://www.digitalpreservation.gov/ndsa/working_groups/documents/NDSA_USWebArchivingSurvey_2013.pdf. A report of the National Digital Stewardship Alliance. [Accessed: 11/01/2016].

[26] Daniel Gomes, João Miranda, and Miguel Costa. A Survey on Web Archiving Initiatives. In *Research and Advanced Technology for Digital Libraries*, pages 408–420. Springer Berlin Heidelberg, 2011. doi: 10.1007/978-3-642-24469-8_41.

[27] Susan Schreibman, Ray Siemens, and John Unsworth. *A Companion to Digital Humanities*. Blackwell Publishing, 2008.

[28] J.-B. Michel, Y. K. Shen, A. P. Aiden, A. Veres, M. K. Gray, J. P. Pickett, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant, S. Pinker, M. A. Nowak, and E. L. Aiden and. Quantitative Analysis of Culture Using Millions of Digitized Books. *Science*, 331(6014):176–182, dec 2010. doi: 10.1126/science.1199644.

[29] Sarah Cohen, Chengkai Li, Jun Yang, and Cong Yu. Computational Journalism: A Call to Arms to Database Researchers. In *Proceedings of the 5th Biennial Conference on Innovative Data Systems Research*, pages 148–151, 2011.

[30] Helen Hockx-Yu. Access and Scholarly Use of Web Archives. *Alexandria: The Journal of National and International Library and Information Issues*, 25(1-2):113–127, 2014. doi: 10.7227/alx.0023.

[31] Franco Moretti. *Graphs, Maps, Trees: Abstract Models for a Literary History*. Verso, 2005.

[32] Vinay Goel. Beta Wayback Machine - Now with Site Search!, oct 2016. URL https://blog.archive.org/2016/10/24/beta-wayback-machine-now-with-site-search. [Accessed: 16/03/2017].

[33] Nikos Kasioumis, Vangelis Banos, and Hendrik Kalb. Towards Building a Blog Preservation Platform. *World Wide Web Journal*, 17(4):799–825, 2014. doi: 10.1007/s11280-013-0234-4.

[34] Catherine C Marshall and Frank M Shipman. An Argument for Archiving Facebook As a Heterogeneous Personal Store. In *Proceedings of the 14th ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '14*, pages 11–20. IEEE Press, 2014. doi: 10.1109/jcdl.2014.6970144.

[35] Hany M. SalahEldeen and Michael L. Nelson. Losing My Revolution: How Many Resources Shared on Social Media Have Been Lost? In *Theory and Practice of Digital Libraries*, TPDL'12, pages 125–137, Paphos, Cyprus, 2012. Springer. doi: 10.1007/978-3-642-33290-6_14.

[36] Nattiya Kanhabua, Roi Blanco, Kjetil Nørvåg, et al. Temporal Information Retrieval. *Foundations and Trends® in Information Retrieval*, 9(2):91–208, 2015. doi: 10.1145/2911451.2914805.

[37] Omar Alonso, Michael Gertz, and Ricardo Baeza-Yates. On the Value of Temporal Information in Information Retrieval. *ACM SIGIR Forum*, 41(2): 35–41, dec 2007. doi: 10.1145/1328964.1328968.

[38] Ricardo Campos, Gaël Dias, Alípio M Jorge, and Adam Jatowt. Survey of Temporal Information Retrieval and Related Applications. *ACM Computing Surveys (CSUR)*, 47(2):15, 2015.

[39] Rosie Jones and Fernando Diaz. Temporal Profiles of Queries. *ACM Transactions on Information Systems*, 25(3):14–es, jul 2007. doi: 10.1145/1247715.1247720.

[40] Klaus Berberich, Srikanta Bedathur, Omar Alonso, and Gerhard Weikum. A Language Modeling Approach for Temporal Information Needs. In *Proceedings of the 32Nd European Conference on Advances in Information Retrieval*

(ECIR), ECIR'2010, pages 13–25, Berlin, Heidelberg, 2010. Springer-Verlag. doi: 10.1007/978-3-642-12275-0_5.

[41] Jaspreet Singh, Wolfgang Nejdl, and Avishek Anand. History by Diversity: Helping Historians Search News Archives. In *Proceedings of the 2016 ACM on Conference on Human Information Interaction and Retrieval - CHIIR '16*, pages 183–192. ACM Press, 2016. doi: 10.1145/2854946.2854959.

[42] Avishek Anand, Srikanta Bedathur, Klaus Berberich, and Ralf Schenkel. Temporal Index Sharding for Space-time Efficiency in Archive Search. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '11*, pages 545–554, New York, NY, USA, 2011. ACM. doi: 10.1145/2009916.2009991.

[43] Avishek Anand, Srikanta Bedathur, Klaus Berberich, and Ralf Schenkel. Index Maintenance for Time-travel Text Search. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '12*, pages 235–244, Portland, Oregon, USA, 2012. doi: 10.1145/2348283.2348318.

[44] Andrei Broder. A Taxonomy of Web Search. In *ACM Sigir forum*, volume 36, pages 3–10. ACM, Association for Computing Machinery (ACM), sep 2002. doi: 10.1145/792550.792552.

[45] Miguel Costa and Mário J Silva. Understanding the Information Needs of Web Archive Users. In *Proceedings of the 10th International Web Archiving Workshop*, 2010.

[46] Miguel Costa, Daniel Gomes, Francisco Couto, and Mário Silva. A Survey of Web Archive Search Architectures. In *Proceedings of the 22nd International Conference on World Wide Web - WWW '13 Companion*, pages 1045–1050, New York, NY, USA, 2013. ACM Press. doi: 10.1145/2487788.2488116.

[47] Nattiya Kanhabua, Philipp Kemkes, Wolfgang Nejdl, Tu Ngoc Nguyen, Felipe Reis, and Nam Khanh Tran. How to Search the Internet Archive Without Indexing It. In *Research and Advanced Technology for Digital Libraries*, pages 147–160, Hannover, Germany, 2016. Springer International Publishing. doi: 10.1007/978-3-319-43997-6_12.

[48] Miroslav Shaltev, Jan-Hendrik Zab, Philipp Kemkes, Stefan Siersdorfer, and Sergej Zerr. Cobwebs from the Past and Present: Extracting Large Social Networks Using Internet Archive Data. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '16*, Pisa, Italy, 2016. doi: 10.1145/2911451.2911467.

[49] Wendy Hall, Jim Hendler, and Steffen Staab. A Manifesto for Web Science @10. *arXiv:1702.08291*, 2017.

[50] J. Cho and H. Garcia-Molina. The Evolution of the Web and Implications for an Incremental Crawler. In *Proceedings of the 26th International Conference on Very Large Data Bases*, VLDB '00, 2000.

[51] Dennis Fetterly, Mark Manasse, Marc Najork, and Janet Wiener. A Large-scale Study of the Evolution of Web Pages. In *Proceedings of the 12th International Conference on World Wide Web - WWW '03*, pages 669–678, 2003. doi: 10.1002/spe.577.

[52] Wallace Koehler. Web Page Change and Persistence-a Four-year Longitudinal Study. *Journal of the American Society for Information Science and Technology*, 53(2):162–171, jan 2002. doi: 10.1002/asi.10018.

[53] Alexandros Ntoulas, Junghoo Cho, and Christopher Olston. What's New on the Web?: The Evolution of the Web from a Search Engine Perspective. In *Proceedings of the 13th Conference on World Wide Web - WWW '04*, pages 1–12. ACM Press, 2004. doi: 10.1145/988672.988674.

[54] Eytan Adar, Jaime Teevan, Susan T. Dumais, and Jonathan L. Elsas. The Web Changes Everything: Understanding the Dynamics of Web Content. In *Proceedings of the 2nd ACM International Conference on Web Search and Data Mining - WSDM '09*, pages 282–291. ACM Press, 2009. doi: 10.1145/1498759.1498837.

[55] Scott A. Hale, Taha Yasseri, Josh Cowls, Eric T. Meyer, Ralph Schroeder, and Helen Margetts. Mapping the UK Webspace: Fifteen Years of British Universities on the Web. In *Proceedings of the 2014 ACM Conference on Web Science - WebSci '14*, WebSci '14. ACM Press, 2014. doi: 10.1145/2615569.2615691.

[56] Teru Agata, Yosuke Miyata, Emi Ishita, Atsushi Ikeuchi, and Shuichi Ueda. Life Span of Web Pages: A Survey of 10 Million Pages Collected in 2001. *Digital Libraries*, pages 463–464, 2014. doi: 10.1109/JCDL.2014.6970226.

[57] Lulwah M Alkwai, Michael L Nelson, and Michele C Weigle. How Well Are Arabic Websites Archived? In *Proceedings of the 15th ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '15*, pages 223–232. ACM, 2015. doi: 10.1145/2756406.2756912.

[58] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: A Flexible Data Processing Tool. *Communications of the ACM*, 53(1):72–77, 2010. doi: 10.1145/1629175.1629198.

[59] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark: Cluster Computing with Working Sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, volume 10, page 10, 2010.

[60] Ahmed AlSum. *Web Archive Services Framework for Tighter Integration between the Past and Present Web.* PhD thesis, Old Dominion University, 2014.

[61] Avinash Lakshman and Prashant Malik. Cassandra: A Decentralized Structured Storage System. *ACM SIGOPS Operating Systems Review*, 44(2):35–40, 2010. doi: 10.1145/1773912.1773922.

[62] Jimmy Lin, Milad Gholami, and Jinfeng Rao. Infrastructure for Supporting Exploration and Discovery in Web Archives. In *Proceedings of the 23rd International Conference on World Wide Web - WWW '14 Companion.* ACM Press, 2014. doi: 10.1145/2567948.2579045.

[63] Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C Hsieh, Deborah A Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, and Robert E Gruber. Bigtable: A Distributed Storage System for Structured Data. *ACM Transactions on Computer Systems (TOCS)*, 26(2):4, 2008. doi: 10.1145/1365815.1365816.

[64] Nick Craswell, David Hawking, and Stephen Robertson. Effective Site Finding Using Link Anchor Information. In *Proceedings of the 24th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '01.* ACM, ACM Press, 2001. doi: 10.1145/383952.383999.

[65] Wessel Kraaij, Thijs Westerveld, and Djoerd Hiemstra. The Importance of Prior Probabilities for Entry Page Search. In *Proceedings of the 25th annual international ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '02.* ACM, 2002. doi: 10.1145/564376.564383.

[66] Paul Ogilvie and Jamie Callan. Combining Document Representations for Known-item Search. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval - SIGIR '03.* ACM, ACM Press, 2003. doi: 10.1145/860435.860463.

[67] Marijn Koolen and Jaap Kamps. The Importance of Anchor Text for Ad Hoc Search Revisited. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '10*, pages 122–129. ACM, ACM Press, 2010. doi: 10.1145/1835449.1835472.

[68] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank Citation Ranking: Bringing Order to the Web. *InfoLab*, 1999.

[69] Réka Albert, Hawoong Jeong, and Albert-László Barabási. Internet: Diameter of the World-Wide Web. *Nature*, 401(6749):130–131, sep 1999. doi: 10.1038/43601.

[70] Andrei Broder, Ravi Kumar, Farzin Maghoul, Prabhakar Raghavan, Sridhar Rajagopalan, Raymie Stata, Andrew Tomkins, and Janet Wiener. Graph

Structure in the Web. *Computer Networks*, 33(1):309–320, 2000. doi: 10.1016/s1389-1286(00)00083-9.

[71] Lada A. Adamic, Bernardo A. Huberman, A.-L. Barabási, R. Albert, H. Jeong, and G. Bianconi. Power-Law Distribution of the World Wide Web. *Science*, 287(5461):2115, mar 2000. doi: 10.1126/science.287.5461.2115a.

[72] Torsten Suel and Jun Yuan. Compressing the Graph Structure of the Web. In *Data Compression Conference*, 2001.

[73] P. Boldi and S. Vigna. The WebGraph Framework I: Compression Techniques. In *Proceedings of the 13th Conference on World Wide Web - WWW '04*, pages 595–602, Manhattan, USA, 2004. ACM, ACM Press. doi: 10.1145/988672.988752. URL http://law.di.unimi.it/datasets.php.

[74] Bernardo A Huberman and Lada A Adamic. Internet: Growth Dynamics of the World-wide Web. *Nature*, 401(6749):131–131, 1999.

[75] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graphs over Time: Densification Laws, Shrinking Diameters and Possible Explanations. In *Proceeding of the 1th ACM SIGKDD international conference on Knowledge discovery in data mining - KDD '05*, pages 177–187. ACM, ACM Press, 2005. doi: 10.1145/1081870.1081893.

[76] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph Evolution: Densification and Shrinking Diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):2, mar 2007. doi: 10.1145/1217299.1217301.

[77] Tu Ngoc Nguyen, Nattiya Kanhabua, Claudia Niederée, and Xiaofei Zhu. A Time-aware Random Walk Model for Finding Important Documents in Web Archives. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '15*, pages 915–918, New York, NY, USA, 2015. ACM. doi: 10.1145/2766462.2767832.

[78] Klaus Berberich and Srikanta Bedathur. Temporal Diversification of Search Results. In *SIGIR 2013 Workshop on Time-aware Information Access (TAIA 2013)*, 2013.

[79] Nattiya Kanhabua and Wolfgang Nejdl. On the Value of Temporal Anchor Texts in Wikipedia. In *SIGIR 2014 Workshop on Temporal, Social and Spatiallyaware Information Access (TAIA)*, 2014.

[80] Kerstin Bischoff, Claudiu S. Firan, Wolfgang Nejdl, and Raluca Paiu. Can All Tags Be Used for Search? In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pages 193–202, 2008. doi: 10.1145/1458082.1458112.

[81] Paul Heymann, Georgia Koutrika, and Hector Garcia-Molina. Can Social Bookmarking Improve Web Search? In *Proceedings of the International Conference on Web Search and Data Mining - WSDM '08*. ACM Press, 2008. doi: 10.1145/1341531.1341558.

[82] Raluca Paiu. *Exploiting Tag Information for Search and Personalization.* PhD thesis, Leibniz Universität Hannover, 2009.

[83] Arkaitz Zubiaga, Victor Fresno, Raquel Martinez, and Alberto Perez Garcia-Plaza. Harnessing Folksonomies to Produce a Social Classification of Resources. *IEEE Transactions on Knowledge and Data Engineering*, 25(8): 1801–1813, 2013. doi: 10.1109/tkde.2012.115.

[84] Avishek Anand, Srikanta Bedathur, Klaus Berberich, and Ralf Schenkel. Efficient Temporal Keyword Search Over Versioned Text. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management - CIKM '10*, pages 699–708. ACM Press, 2010. doi: 10.1145/1871437.1871528.

[85] ISO. 26324: 2012 Information and Documentation-digital Object Identifier System, 2012.

[86] Heather A. Piwowar, Roger S. Day, and Douglas B. Fridsma. Sharing Detailed Research Data Is Associated with Increased Citation Rate. *PLoS ONE*, 2(3): 1–5, mar 2007. doi: 10.1371/journal.pone.0000308.

[87] Andrew Treloar. The Research Data Alliance: Globally Co-ordinated Action against Barriers to Data Publishing and Sharing. *Learned Publishing*, 27, 2014. doi: 10.1087/20140503.

[88] Angelina Kraft, Matthias Razum, Jan Potthoff, Andrea Porzel, Thomas Engel, Frank Lange, Karina van den Broek, and Filipe Furtado. The RADAR Project - A Service for Research Data Archival and Publication. *ISPRS International Journal of Geo-Information*, 5(3):28, 2016. doi: 10.3390/ijgi5030028.

[89] Angelina Kraft, Jan Potthoff, and Matthias Razum. Establishing a Generic Research Data Repository. In *Proceedings of the 13th International Conference on Digital Preservation, iPRES 2016*, 2016.

[90] Catherine C. Marshall and Frank M. Shipman. On the Institutional Archiving of Social Media. In *Proceedings of the 12th ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '12*, JCDL '12, pages 1–10, New York, NY, USA, 2012. ACM Press. doi: 10.1145/2232817.2232819.

[91] Siân E. Lindley, Catherine C. Marshall, Richard Banks, Abigail Sellen, and Tim Regan. Rethinking the Web As a Personal Archive. In *Proceedings of the 22nd International Conference on World Wide Web - WWW '13*, pages 749–760, New York, NY, USA, 2013. ACM. doi: 10.1145/2488388.2488454.

[92] Greg Wilson, D. A. Aruliah, C. Titus Brown, Neil P. Chue Hong, Matt Davis, Richard T. Guy, Steven H. D. Haddock, Kathryn D. Huff, Ian M. Mitchell, Mark D. Plumbley, Ben Waugh, Ethan P. White, and Paul Wilson. Best Practices for Scientific Computing. *PLoS Biology*, 12(1), jan 2014. doi: 10.1371/journal.pbio.1001745.

[93] Simon Hettrick, Mario Antonioletti, Les Carr, Neil Chue Hong, Stephen Crouch, David De Roure, Iain Emsley, Carole Goble, Alexander Hay, Devasena Inupakutika, Mike Jackson, Aleksandra Nenadic, Tim Parkinson, Mark I Parsons, Aleksandra Pawlik, Giacomo Peru, Arno Proeme, John Robinson, and Shoaib Sufi. Uk Research Software Survey 2014, dec 2014.

[94] Stephen Crouch, Neil Chue Hong, Simon Hettrick, Mike Jackson, Aleksandra Pawlik, Shoaib Sufi, Les Carr, David De Roure, Carole A. Goble, and Mark Parsons. The Software Sustainability Institute: Changing Research Software Attitudes and Practices. *Computing in Science and Engineering*, 15(6):74–80, 2013. doi: 10.1109/MCSE.2013.133.

[95] Roberto Di Cosmo and Stefano Zacchiroli. Software Heritage: Why and How to Preserve Software Source Code. In *iPRES 2017: 14th International Conference on Digital Preservation*, Kyoto, Japan, sep 2017.

[96] Scott G. Ainsworth, Ahmed Alsum, Hany SalahEldeen, Michele C. Weigle, and Michael L. Nelson. How Much of the Web Is Archived? In *Proceeding of the 11th Annual International ACM/IEEE Joint Conference on Digital Libraries - JCDL '11*, pages 133–136. ACM, ACM Press, 2011. doi: 10.1145/1998076.1998100.

[97] Martin Klein, Herbert Van de Sompel, Robert Sanderson, Harihar Shankar, Lyudmila Balakireva, Ke Zhou, and Richard Tobin. Scholarly Context Not Found: One in Five Articles Suffers from Reference Rot. *PLoS ONE*, 9(12): 1–39, dec 2014. doi: 10.1371/journal.pone.0115253.

[98] Ke Zhou, Claire Grover, Martin Klein, and Richard Tobin. No More 404s: Predicting Referenced Link Rot in Scholarly Articles for Pro-active Archiving. In *Proceedings of the 15th ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '15*, pages 233–236. ACM Press, 2015. doi: 10.1145/2756406.2756940.

[99] Daniel Gomes and Miguel Costa. The Importance of Web Archives for Humanities. *International Journal of Humanities and Arts Computing*, 8(1): 106–123, 2014. doi: 10.3366/ijhac.2014.0122.

[100] Ahmed AlSum, Michele C Weigle, Michael L Nelson, and Herbert Van de Sompel. Profiling Web Archive Coverage for Top-level Domain and Content Language. *International Journal on Digital Libraries*, 14(3-4):149–166, 2014. doi: 10.1007/s00799-014-0118-y.

[101] Michael Day, Ann MacDonald, Maureen Pennock, and Akiko Kimura. Implementing Digital Preservation Strategy: Developing content collection profiles at the British Library. In *Proceedings of the 14th ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '14*, pages 21–24. IEEE, IEEE, sep 2014. doi: 10.1109/jcdl.2014.6970145.

[102] Sawood Alam, Michael L. Nelson, Herbert Van de Sompel, Lyudmila L. Balakireva, Harihar Shankar, and David S. H. Rosenthal. Web Archive Profiling Through CDX Summarization. In *Research and Advanced Technology for Digital Libraries*, pages 3–14. Springer International Publishing, 2015. doi: 10.1007/978-3-319-24592-8_1.

[103] Heinz Pampel, Paul Vierkant, Frank Scholze, Roland Bertelmann, Maxi Kindling, Jens Klump, Hans-Jürgen Goebelbecker, Jens Gundlach, Peter Schirmbacher, and Uwe Dierolf. Making Research Data Repositories Visible: The Re3data. Org Registry. *PLoS ONE*, 8(11):e78080, nov 2013. doi: 10.1371/journal.pone.0078080.

[104] Stuart Macdonald. Edinburgh Datashare - a Dspace Data Repository: Achievements and Aspirations, 2009. URL https://www.era.lib.ed.ac.uk/handle/1842/3201.

[105] R. D. Peng. Reproducible Research in Computational Science. *Science*, 334 (6060):1226–1227, dec 2011. doi: 10.1126/science.1213847.

[106] Thomas Vogt. Software Dokumentieren! *Mitteilungen der Deutschen Mathematiker-Vereinigung*, 22(1):16–17, 2014. doi: 10.1515/dmvm-2014-0011.

[107] Natalie J Stanford, Katherine Wolstencroft, Martin Golebiewski, Renate Kania, Nick Juty, Christopher Tomlinson, Stuart Owen, Sarah Butcher, Henning Hermjakob, Nicolas Le Novère, et al. The Evolution of Standards and Data Management Practices in Systems Biology. *Molecular systems biology*, 11 (12), 2015. doi: 10.15252/msb.20156053.

[108] Gert-Martin Greuel and Wolfram Sperber. swMATH – an information service for mathematical software. In *Mathematical Software – ICMS 2014*, pages 691–701. Springer Berlin Heidelberg, 2014. doi: 10.1007/978-3-662-44199-2_103.

[109] Michael Kohlhase and Wolfram Sperber. Software Citations, Information Systems, and Beyond. In *Lecture Notes in Computer Science*, pages 99–114. Springer International Publishing, 2017. doi: 10.1007/978-3-319-62075-6_8.

[110] Marc Spaniol, Arturas Mazeika, Dimitar Denev, and Gerhard Weikum. “catch Me If You Can”: Visual Analysis of Coherence Defects in Web Archiving. In *The 9th International Web Archiving Workshop (IWAW 2009) Corfu, Greece*, page 1, 2009.

[111] Sagemath. http://tempas.l3s.de/micrawler/permalink/8bcbcec, 2018. Archived using Micrawler: 2018-01-10T09:03:35.000Z.

[112] Ricardo Baeza-Yates and Barbara Poblete. Dynamics of the Chilean Web Structure. *Computer Networks*, 50(10):1464–1473, 2006. doi: 10.1016/j.comnet.2005.10.017.

[113] Ilaria Bordino, Paolo Boldi, Debora Donato, Massimo Santini, and Sebastiano Vigna. Temporal Evolution of the UK Web. In *2008 IEEE International Conference on Data Mining Workshops*, pages 909–918. IEEE, dec 2008. doi: 10.1109/icdmw.2008.88.

[114] Annika Hinze, Craig Taube-Schock, David Bainbridge, Rangi Matamua, and J Stephen Downie. Improving Access to Large-scale Digital Libraries Throughsemantic-enhanced Search and Disambiguation. In *Proceedings of the 15th ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '15*, 2015. doi: 10.1145/2756406.2756920.

[115] Jimmy Lin. Warcbase on Github, 2013-2017. URL https://github.com/lintool/warcbase. [Accessed: 11/01/2016].

[116] Kathryn Schulz. What Is Distant Reading. *The New York Times*, 24, 2011.

[117] Franco Moretti. *Distant Reading*. Verso Books, 2013.

[118] Stefan Jänicke, Greta Franzini, Muhammad Faisal Cheema, and Gerik Scheuermann. On Close and Distant Reading in Digital Humanities: A Survey and Future Challenges. In R. Borgo, F. Ganovelli, and I. Viola, editors, *Eurographics Conference on Visualization (EuroVis) - STARs*. The Eurographics Association, 2015. doi: 10.2312/eurovisstar.20151113.

[119] Emily Maemura, Christoph Becker, and Ian Milligan. Understanding Computational Web Archives Research Methods Using Research Objects. In *2016 IEEE International Conference on Big Data (Big Data)*. IEEE, dec 2016. doi: 10.1109/bigdata.2016.7840982.

[120] Jeffrey S. Saltz. The Need for New Processes, Methodologies and Tools to Support Big Data Teams and Improve Big Data Project Effectiveness. In *2015 IEEE International Conference on Big Data (Big Data)*. IEEE, oct 2015. doi: 10.1109/bigdata.2015.7363988.

[121] Anirudh Kadadi, Rajeev Agrawal, Christopher Nyamful, and Rahman Atiq. Challenges of Data Integration and Interoperability in Big Data. In *2014 IEEE International Conference on Big Data (Big Data)*. IEEE, oct 2014. doi: 10.1109/bigdata.2014.7004486.

[122] Niels Brügger. Web History, Web Archives, and Web Research Infrastructure - between Close and Distant Reading, 2015. URL http://alexandria-project.eu/events/2nd-int-alexandria-workshop-2015. Keynote at

the 2nd Int. Alexandria Workshop on Foundations for Temporal Retrieval, Exploration and Analytics in Web Archives on 03/11/2015 [Accessed: 17/01/2016].

[123] Destatis) German Federal Statistical Office (Statistisches Bundesamt. Fast Zehn Jahre Euro - Preisentwicklung Vor Und Nach Der Bargeldeinführung, dec 2011. URL `https://www.destatis.de/DE/Publikationen/Thematisch/Preise/Verbraucherpreise/Fast10JahreEuro5611105119004.html`. [Accessed: 16/03/2017].

[124] Dan Brickley, Ramanathan V Guha, and Brian McBride. RDF Schema 1.1. *W3C Recommendation*, 2014.

[125] Eric Prud'hommeaux, Andy Seaborne, et al. Sparql Query Language for RDF. *W3C Recommendation*, 15, 2008.

[126] Eric Prud'hommeaux, Carlos Buil-Aranda, et al. S1.1 Federated Query. *W3C Recommendation*, 21, 2013.

[127] Tom Heath and Christian Bizer. Linked Data: Evolving the Web into a Global Data Space. *Synthesis Lectures on the Semantic Web: Theory and Technology*, 1(1):1–136, feb 2011. doi: 10.2200/s00334ed1v01y201102wbe001.

[128] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, et al. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*, 6(2):167–195, 2015.

[129] Kevin R. Page, Sean Bechhofer, Gyorgy Fazekas, David M. Weigl, and Thomas Wilmering. Realising a Layered Digital Library: Exploration and Analysis of the Live Music Archive through Linked Data. In *2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*. IEEE, IEEE, jun 2017. doi: 10.1109/jcdl.2017.7991563.

[130] Sébastien Ferré. Sparklis: a SPARQL Endpoint Explorer for Expressive Question Answering. In *ISWC Posters & Demonstrations Track*, Riva del Garda, Italy, oct 2014.

[131] Marcelo Arenas, Bernardo Cuenca Grau, Evgeny Kharlamov, Sarunas Marciuska, Dmitriy Zheleznyakov, and Ernesto Jimenez-Ruiz. Semfacet: Semantic Faceted Search Over Yago. In *Proceedings of the 23rd International Conference on World Wide Web - WWW '14 Companion*. ACM, ACM Press, 2014. doi: 10.1145/2567948.2577011.

[132] Giovanni Maria Sacco and Yannis Tzitzikas. *Dynamic Taxonomies and Faceted Search: Theory, Practice, and Experience*, volume 25. Springer Science & Business Media, 2009.

[133] Yannis Tzitzikas, Nikos Manolis, and Panagiotis Papadakos. Faceted Exploration of Rdf/s Datasets: A Survey. *Journal of Intelligent Information Systems*, pages 1–36, 2016. doi: 10.1007/s10844-016-0413-8.

[134] Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. Template-based question answering over RDF data. In *Proceedings of the 21st International Conference on World Wide Web - WWW '12*. ACM, ACM Press, 2012. doi: 10.1145/2187836.2187923.

[135] Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. Fast and Space-efficient Entity Linking in Queries. In *Proceedings of the 8th ACM International Conference on Web Search and Data Mining - WSDM '15*, New York, NY, USA, 2015. ACM. doi: 10.1145/2684822.2685317.

[136] Pavlos Fafalios, Vaibhav Kasturia, and Wolfgang Nejdl. Towards a Ranking Model for Semantic Layers over Digital Archives. In *2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, Toronto, Ontario, Canada, jun 2017. IEEE. doi: 10.1109/jcdl.2017.7991617.

[137] Pavlos Fafalios, Vaibhav Kasturia, and Wolfgang Nejdl. Ranking Archived Documents for Structured Queries on Semantic Layers. In *Proceedings of the 18th ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '18*, 2018.

[138] Minas Gjoka, Maciej Kurant, Carter T Butts, and Athina Markopoulou. Practical Recommendations on Crawling Online Social Networks. *IEEE Journal on Selected Areas in Communications*, 29(9):1872–1892, 2011. doi: 10.1109/jsac.2011.111011.

[139] Rong-Hua Li, Jeffrey Xu Yu, Lu Qin, Rui Mao, and Tan Jin. On Random Walk Based Graph Sampling. In *2015 IEEE 31st International Conference on Data Engineering*, pages 927–938. IEEE, IEEE, apr 2015. doi: 10.1109/icde.2015.7113345.

[140] Miguel Costa, Daniel Gomes, and Mário J. Silva. The Evolution of Web Archiving. *International Journal on Digital Libraries*, 18(3):191–205, may 2016. doi: 10.1007/s00799-016-0171-9.

[141] Daniel Gomes, Sérgio Freitas, and Mário J. Silva. Design and Selection Criteria for a National Web Archive. In *Research and Advanced Technology for Digital Libraries*, pages 196–207, 2006. doi: 10.1007/11863878_17.

[142] Ahmed Alsum, Michele C. Weigle, Michael L. Nelson, and Herbert Van de Sompel. Profiling Web Archive Coverage for Top-level Domain and Content Language. In *Research and Advanced Technology for Digital Libraries*, pages 60–71, 2013. doi: 10.1007/978-3-642-40501-3_7.

[143] Hugo C. Huurdeman, Anat Ben-David, Jaap Kamps, Thaer Samar, and Arjen P. de Vries. Finding Pages on the Unarchived Web. In *Proceedings of the 14th ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '14*. IEEE, sep 2014. doi: 10.1109/jcdl.2014.6970188.

[144] Sawood Alam, Michael L. Nelson, Herbert Van de Sompel, and David S. H. Rosenthal. Web Archive Profiling through Fulltext Search. In *Research and Advanced Technology for Digital Libraries*, pages 121–132. Springer International Publishing, 2016. doi: 10.1007/978-3-319-43997-6_10.

[145] Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. Combating Web Spam with TrustRank. In *Proceedings 2004 VLDB Conference*, pages 576–587. VLDB Endowment, Elsevier, 2004. doi: 10.1016/b978-012088469-8.50052-8.

[146] Jon M. Kleinberg. Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM*, 46(5):604–632, sep 1999. doi: 10.1145/324133.324140.

[147] Taher H Haveliwala. Topic-sensitive Pagerank. In *Proceedings of the 11th International Conference on World Wide Web - WWW '02*, pages 517–526. ACM, 2002. doi: 10.1145/511446.511513.

[148] Andrew Y. Ng, Alice X. Zheng, and Michael I. Jordan. Link Analysis, Eigenvectors and Stability. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'01, pages 903–910, 2001.

[149] Paolo Boldi, Massimo Santini, and Sebastiano Vigna. Do Your Worst to Make the Best: Paradoxical Effects in PageRank Incremental Computations. In *International Workshop on Algorithms and Models for the Web-Graph*, pages 168–180. Springer, Springer Berlin Heidelberg, 2004. doi: 10.1007/978-3-540-30216-2_14.

[150] Andrea Vattani, Deepayan Chakrabarti, and Maxim Gurevich. Preserving Personalized Pagerank in Subgraphs. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 793–800, 2011.

[151] Jure Leskovec and Christos Faloutsos. Sampling from Large Graphs. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '06*, pages 631–636. ACM, ACM Press, 2006. doi: 10.1145/1150402.1150479.

[152] Tianyi Wang, Yang Chen, Zengbin Zhang, Peng Sun, Beixing Deng, and Xing Li. Unbiased Sampling in Directed Social Graph. In *ACM SIGCOMM Computer Communication Review*, volume 40, pages 401–402. ACM, 2010. doi: 10.1145/1851182.1851231.

[153] Arun S. Maiya and Tanya Y. Berger-Wolf. Benefits of Bias: Towards Better Characterization of Network Sampling. In *Proceedings of the 17th ACM*

SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’11, pages 105–113, San Diego, California, USA, 2011. ACM. doi: 10.1145/2020408.2020431.

[154] Christian Hübler, Hans-Peter Kriegel, Karsten Borgwardt, and Zoubin Ghahramani. Metropolis Algorithms for Representative Subgraph Sampling. In *2008 Eighth IEEE International Conference on Data Mining*, pages 283–292. IEEE, IEEE, dec 2008. doi: 10.1109/icdm.2008.124.

[155] Zhuojie Zhou, Nan Zhang, Zhiguo Gong, and Gautam Das. Faster Random Walks by Rewiring Online Social Networks On-the-fly. *ACM Transactions on Database Systems (TODS)*, 40(4):26, 2016. doi: 10.1145/2847526.

[156] Stephen J Hardiman and Liran Katzir. Estimating Clustering Coefficients and Size of Social Networks Via Random Walk. In *Proceedings of the 22nd International Conference on World Wide Web - WWW '13*, pages 539–550. ACM, 2013. doi: 10.1145/2488388.2488436.

[157] Liran Katzir, Edo Liberty, and Oren Somekh. Estimating Sizes of Social Networks Via Biased Sampling. In *Proceedings of the 20th International Conference on World Wide Web - WWW '11*, pages 597–606. ACM, ACM Press, 2011. doi: 10.1145/1963405.1963489.

[158] Anirban Dasgupta, Ravi Kumar, and Tamas Sarlos. On Estimating the Average Degree. In *Proceedings of the 23rd International Conference on World Wide Web - WWW '14*, pages 795–806. ACM, ACM Press, 2014. doi: 10.1145/2566486.2568019.

[159] Maurice G Kendall. A New Measure of Rank Correlation. *Biometrika*, 30 (1/2):81–93, 1938. doi: 10.2307/2332226.

[160] Reynold S. Xin, Joseph E. Gonzalez, Michael J. Franklin, and Ion Stoica. Graphx: A Resilient Distributed Graph System on Spark. In *First International Workshop on Graph Data Management Experiences and Systems - GRADES '13*, GRADES ’13. ACM Press, 2013. doi: 10.1145/2484425.2484427.

[161] Paolo Boldi, Marco Rosa, Massimo Santini, and Sebastiano Vigna. Layered Label Propagation: A Multiresolution Coordinate-free Ordering for Compressing Social Networks. In Sadagopan Srinivasan, Krithi Ramamritham, Arun Kumar, M. P. Ravindra, Elisa Bertino, and Ravi Kumar, editors, *Proceedings of the 20th International Conference on World Wide Web - WWW '11*, pages 587–596. ACM Press, 2011. doi: 10.1145/1963405.1963488.

[162] Archiveteam. Friendster Social Network Dataset: Friends, 2011. URL `https://archive.org/details/friendster-dataset-201107`. published under CC0 1.0 Universal.

[163] Béla Bollobás, Christian Borgs, Jennifer Chayes, and Oliver Riordan. Directed Scale-free Graphs. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '03, Baltimore, Maryland, 2003.

[164] Andrew Jackson, Jimmy Lin, Ian Milligan, and Nick Ruest. Desiderata for Exploratory Search Interfaces to Web Archives in Support of Scholarly Activities. In *Proceedings of the 16th ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL '16*, pages 103–106, Newark, New Jersey, USA, 2016. IEEE, ACM Press. doi: 10.1145/2910896.2910912.

[165] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schutze. *Introduction to Information Retrieval.* Cambridge University Press, 2008. doi: 10.1017/cbo9780511809071.

[166] Miguel Costa, Francisco Couto, and Mário Silva. Learning Temporal-dependent Ranking Models. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '14*, pages 757–766, New York, NY, USA, 2014. ACM. doi: 10.1145/2600428.2609619.

[167] Steve Bailey and Dave Thompson. Building the Uk's First Public Web Archive. *D-Lib Magazine*, 12(1):1082–9873, 2006.

[168] Daniel Gomes, André Nogueira, João Miranda, and Miguel Costa. Introducing the Portuguese Web Archive Initiative. In *8th International Web Archiving Workshop*. Springer, 2009.