A Survey of Web Archive Search Architectures

Miguel Costa [1,2]
miguel.costa@fccn.pt

Daniel Gomes [1]
daniel.gomes@fccn.pt

Francisco M Couto [2]
fcouto@di.fc.ul.pt

Mário J. Silva [3]
mjs@inesc-id.pt

[1] Foundation for National Scientific Computing, Portugal
[2] University of Lisbon, Faculty of Sciences, LaSIGE, Portugal
[3] University of Lisbon, IST/INESC-ID, Portugal

ABSTRACT

Web archives already hold more than 282 billion documents and users demand full-text search to explore this historical information. This survey provides an overview of web archive search architectures designed for time-travel search, i.e. full-text search on the web within a user-specified time interval. Performance, scalability and ease of management are important aspects to take in consideration when choosing a system architecture. We compare these aspects and initialize the discussion of which search architecture is more suitable for a large-scale web archive.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Search process; H.3.7 [**Digital Libraries**]: Systems issues

General Terms

Web, Archive, Architecture, Search, Preservation

Keywords

Portuguese Web Archive, Temporal Search

1. INTRODUCTION

No one knows the real size of the world wide web. According to Google, in 2008 the web had more than a trillion of unique web pages[1]. In 2010, Google's ex-CEO Eric Schmidt said that we create as much data in two days, around five exabytes, as we did from the dawn of man up until 2003[2]. The fast development of Information and Communication Technology had a great impact on this growth. In the last decade, the world population with access to the Internet grew more than 1 000% in some regions[3]. Computer-based devices and mobile phones with Internet connectivity are now about 5 billion[4], much of which are equipped with technology that empowers people to easily create data. Moreover, software tools such as social networks, blogs and content management systems made it easier for people to publish and share data. This combination of factors resulted in the largest source of information ever created: the world wide web.

However, the web is very dynamic and a large amount of information is lost everyday. Ntoulas et al. found that 80% of the web pages are not available after one year [21]. In a few years they are all likely to disappear, creating a knowledge gap for future generations. Aware of this problem, at least 77 initiatives[5] undertaken by national libraries, national archives and consortia of organizations are archiving parts of the web. Together, these initiatives already hold more than 282 billion documents and this number continues to grow as new initiatives arise.

Full-text search has become the dominant form of finding information on the web, as notoriously seen in web search engines. It gives users the ability to quickly search through vast amounts of unstructured text, powered by sophisticated ranking tools that order results based on how well they match user queries. Following this tendency, full-text search is the users' most desired web archive functionality [22] and the most used when supported [11]. In web archives, collections are not only searchable by text. Web archives also support time-travel queries, i.e. full-text search on the web within a user-specified time interval [9]. This can be considered a *killer application* for web archives, making historical analysis possible and enabling use cases such as, revisiting old news, recovering a missing site or analyzing digital historical documents.

This paper presents an overview of web archive search architectures designed for supporting time-travel queries. Index structures typically used in search engines and optimized for keyword matching handle sub-optimally the queries restricted within a specific period of interest. Moreover, the indexes are not thought to be partitioned in a way that enable web archives to scale and be easily manageable as new collections are added to the system. Given the very long life expectancy of web archives, it is expected that the data size will increase several orders of magnitude, largely surpassing the size indexed by top commercial web search engines. We compare the strategies to partition and distribute these indexes by time in terms of scalability and performance. We also discuss the pros and cons of existent architectures that

[1] `http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html`
[2] `http://techcrunch.com/2010/08/04/schmidt-data/`
[3] `http://www.internetworldstats.com/stats.htm`
[4] `http://www.networkworld.com/news/2010/081610-5billion-devices-internet.html`
[5] `http://en.wikipedia.org/wiki/List_of_Web_archiving_initiatives`

TWAW 2013,, May 13, 2013, Rio de Janeiro, Brazil.
ACM .

face the enormous challenge of making all these data searchable.

The remainder of this paper is organized as follows. In Section 2, we cover the related work that addresses the index structures typically used in search engines and their partitioning by time in Section 3. Section 4 presents the existing web archive architectures, and Section 5 finalizes with the conclusions.

2. RELATED WORK

2.1 Web Archive Search & Analytics

Much of the current effort on web archive development focuses on acquiring, storing, managing and preserving data [18]. However, the data must also be accessible to users who need to exploit and analyze them. Web archives face many challenges related to scalability and information overload, because they accumulate all previous documents and indexes, unlike web search engines that tend to drop old versions when new ones are discovered. Due to the challenge of indexing all the collected data, the prevalent access method in web archives is based on URL search, which returns a list of chronologically ordered versions of that URL. A recent survey reported that 89% of the world-wide web archives support this type of access [14]. However, this type of search is limited, as it forces the users to remember the URLs, some of which refer to content that ceased to exist many years ago. Another type of access is meta-data search, for instance by category or theme, which was shown to be provided by 79% of web archives. Full-text search has become the dominant form of information access, specially in web search systems, such as Google, which has a strong influence on the way users search in other settings. Even with the high computational resources required for this purpose, 67% of world-wide web archives surveyed support full-text search for at least a part of their collections [14]. In another survey about European web archives this percentage is 70% [13].

Several analyses can be performed over web archive data. For instance, an analysis similar to those performed over the 4% of all books ever printed (around 5 million books) to investigate cultural trends quantitatively [20]. This type of analysis provides insights about diverse fields focused in linguistic and cultural phenomena, such as the evolution of grammar or collective memory. Despite not being completely clear how such research can be operationalized, some social problems have already started to be studied. The Yesternet project joined social scientists alongside with computer scientists, to study problems like the diffusion of innovation and beliefs, or the human behavior in social networks [5, 6]. They used the Internet Archive collections since 1996 as main source. The Virtual Observatory for the Study of Online Networks conducted an empirical research in social and political networks over the web to see how they impact the real world and vice-versa [1]. The Living Web Archives (LiWA) aimed to provide contributions to make archived information accessible [19]. It addressed problems shared with other information retrieval (IR) areas, such as web spam detection, terminology evolution, capture of stream video, and assuring temporal coherence of archived content. LiWA was followed by the Longitudinal Analytics of Web Archive data (LAWA), which aims to build an experimental testbed for large-scale data analytics [28]. All these longitudinal studies obtained their research data from web archives.

2.2 Inverted Files

The large size of web collections demands specialized techniques for efficient IR. The inverted index (a.k.a. inverted file) is the most efficient index structure for textual search and can be easily compressed to reduce space [30]. It is composed of two elements: a lexicon and posting lists. The lexicon is the set of all terms that occur on the documents collection. Each term of the lexicon points to a posting list, which is a list of identifiers of documents where the term occurs. Each of these identifiers, id, has usually associated a payload, p, that may include the frequency of the term into the document and information about where the term occurs (e.g. in title). The pair $< id, p >$ is called a posting. For a query to match the potential relevant documents, each query term is searched on the lexicon and the posting list it points to is fetched. The result is the combination of the identifiers on the posting lists.

Web collections continue to grow as new snapshots are crawled and archived. A centralized index for all these data, even if possible, would be inefficient. A distributed approach requires partitioning the index and spreading it by several computers to parallelize searching. Inverted files are mainly partitioned in two ways, as illustrated in Figure 1. They can be partitioned by document or term [30]. Document-based partition (DP) refers to splitting the index per document (vertically). Actually, each computer creates a sub-index of a disjoint subset of documents. A query response is the merging of the results produced by all computers using their sub-indexes. Term-based partition (TP) refers to splitting the index per term (horizontally) and allocating each posting list to a computer. This requires that computers execute pairwise exchanges of posting lists after they create their sub-indexes. A query response is the joining of results produced by the computers that have at least one query term in their sub-indexes. The major differences between both partitions are that:

- in DP, all computers are devoted to the processing of each query, achieving a higher parallelism that reduces response time. In TP, at most one computer per query term responds to a query. The others are free to respond to other requests, achieving a higher concurrency. This results in fewer disk seeks, which is a dominant factor in the search time.
- in DP, all information for matching a document is available locally, avoiding communication between computers. This results in a nearly linear scale out. In TP, the posting lists allocated in different computers must be joined at some point. Usually, a broker joins postings by repeatedly requesting batches of them until it has enough.
- DP does not require rebuilding of the indexes when new documents are indexed, while TP does by adding the new documents to the existent posting lists. Rebuilding the indexes each time a new web collection is integrated into a web archive can be prohibitive.

Usually, commercial web search engines, such as Google, use the document-based partition [8]. Results show that this partition achieves superior query throughput [30].

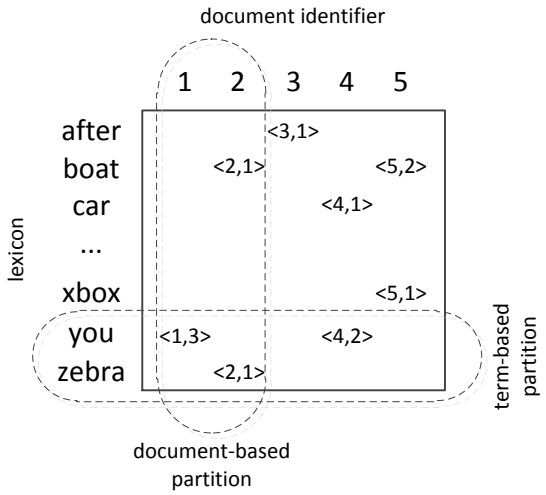

Figure 1: The two main ways to partition an inverted file.

3. TEMPORAL INVERTED FILES

Web archives receive a significant number of queries with a specific time interval of interest, denoted time-travel queries [11]. Thus, partitioning the index by time enables searching only the sub-index corresponding to that interval. In this work, we refer to the time when the web documents were crawled, but we could use other time attributes associated to documents, such as their creation or publication dates. Some works use instead the time mentioned on the document text [2].

We can have at least four types of partitions, where the index is split per time and then per document or term, or the opposite. When partitioning first by time, subsets of documents are allocated to computer clusters according to their timestamps. Then, each subset can have a document-based or term-based partition within the cluster, as illustrated in Figures 2(a) and 2(b), respectively. This offers a simple way to scale web archive indexes, since new clusters are added to the system as new web snapshots are indexed. Another advantage is that time-travel queries will be sent only to the clusters handling that time interval.

When partitioning first by document and then by time, subsets of documents are allocated to computer clusters according to their identifiers (e.g. URLs), and then each subset within a cluster is partitioned according to the timestamps of the documents. Figure 2(c) depicts this partition, where all versions of a document are in the same cluster. In the same way, when partitioning first by term and then by time, subsets of posting lists are allocated to computer clusters according to their terms, and then the postings of each subset are partitioned according to their documents' timestamps. See Figure 2(d). The advantage of these last two partitions is that the sub-indexes of each cluster can be overlapped to reduce space. The sub-indexes have documents that span multiple temporal partitions and are replicated across posting lists. Berberich et al. extended postings with a validity time interval $[t_b, t_e]$, having the form $< id, t_b, t_e, p >$ [9]. On the other hand, the overlapping increases the size of posting lists and consequently the response time of the IR system. It is a trade-off between space and speed that must be chosen accordingly.

3.1 Discussion

The document-based partition of the index offers superior query throughput and does not require rebuilding of the indexes each time new documents are indexed, contrary to the term-based partition. These are two important aspects of decision, specially the last one in the case of a web archive, because rebuilding the indexes involves a huge computational effort and complexity. Imagine the cost of rebuilding the indexes of tens of large-scale web collections each time a new crawl ends. Alternatives for rebuilding the indexes when using the term-based partition exist, but are also more complex and less efficient than using the document-based partition [3]. The decision of partitioning the index first or after time, presents itself as a trade-off between speed and space. However, partitioning first by time has the additional advantage of offering a simple way to manage and scale web archive indexes, since new clusters are added to the system as new web snapshots are indexed. The system can also be implemented with standard inverted indexes, which have received a large amount of research on how to better organize, compress and access such indexes [7].

4. WEB ARCHIVE ARCHITECTURES

The specification of a web archive search architecture comprises the definition of the necessary components to support the service, how these components interact and the requirements governing those interactions. From the user's perspective there are three main requirements: good quality search results, short response times, and a large coverage of the web throughout time. These requirements are key drivers for the architectural design decisions. Other requirements, mostly non-functional, are also important. They include high availability, i.e. how often the system is accessible to users, and high scalability, i.e. the ability to cope with growing amounts of load or data as we add more resources to the system. Given the very long life expectancy of the service, it is expected that the data size will increase several orders of magnitude and that many technological changes will occur. Hence, the architecture must be created without the need to include any special hardware or proprietary soft-

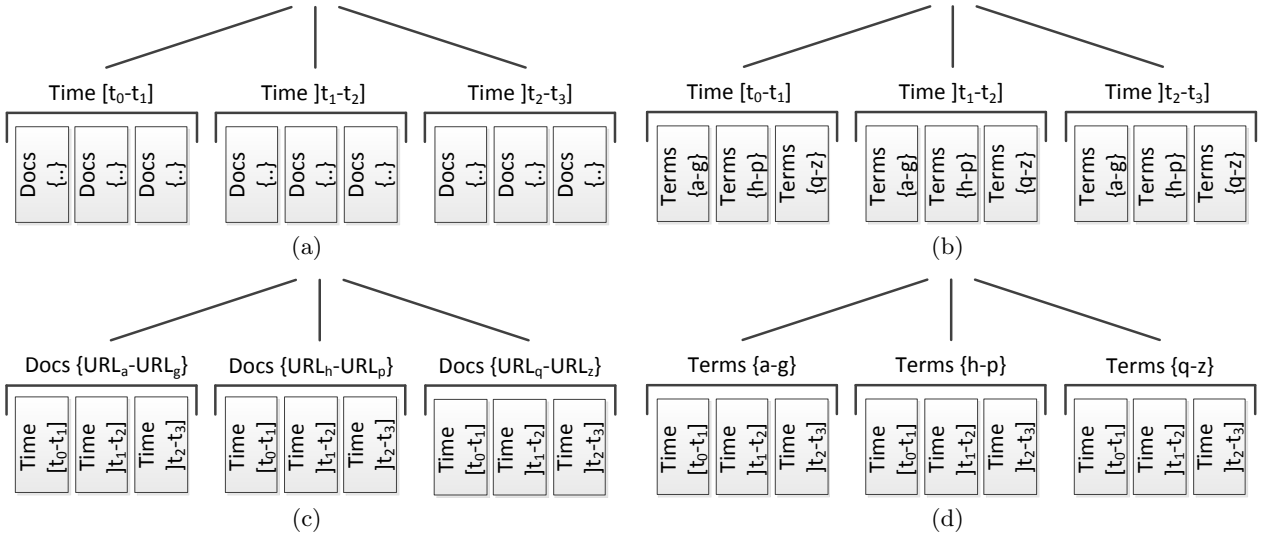

Figure 2: Index partitions using three dimensions: time, documents and terms.

ware. Instead, it should be modular enough to enable the easy exchange of core modules as technology evolves. Next, we present three distinct web archive search architectures that are representative of all surveyed systems [14].

4.1 Wayback Machine

The open source Wayback Machine (WM) is a set of loosely coupled modules that can be exchanged and configured according to the web archive needs [27]. This architecture supports the Internet Archive's WM[6], which serves tens of millions of daily requests, over 500 terabytes of web documents archived since 1996 [17]. Currently, the WM contains more than 150 billion documents from more than 200 million web sites, which makes it the largest web archive in the world. Its search is, however, limited by users having to know beforehand which URL to search.

The Internet Archive's WM uses as index a flat file sorted alphabetically by URL and divided in similar size buckets. The buckets are distributed across web servers and map URLs to the ARC files storing them [10]. Thus, each web server responds over a subset of documents (i.e. URLs), meaning that this architecture uses a document-based index partition. When the WM receives a URL query, a broker routes it to the web server handling that range of URLs, which replies with the list of all URL versions. To access one of these versions, the web server replies with the respective ARC name. Then, the ARC file is located by sending a UDP broadcast to all storage servers. There are more than 2500 storage servers in the primary data center, each with up to four commodity disks. The storage server containing the ARC file replies to the broadcast with the local file path and its identification. Finally, the client browser is redirected to the respective storage server, which runs a lightweight web server to deliver the ARC file.

Since all storage server replicas respond to each broadcast request, a computer failure does not interrupt the service. However, by having all replicas responding to the same request, a lot of work is replicated unnecessarily. The main advantage of this architecture is that it enables the system to be easily extended or replicated, just by adding more computers to the cluster. The current WM is also replicated across three geographically remote sites.

[6] `http://web.archive.org`

4.2 Portuguese Web Archive

The Portuguese Web Archive (PWA)[7] is based on the WM, but uses NutchWAX as its full-text and URL search engine [15]. NutchWAX was developed by the International Internet Preservation Consortium (IIPC) and is used by many web archives [14]. It is an extension of the Nutch search engine with Web Archive eXtensions. There are some older descriptions of Nutch experiments on the Internet Archive [24]. Nutch is a search engine built on top of Lucene [16] that follows a document-based partition of the index. The PWA was extended to partition collections first by time and only then by documents (see Figure 2(a)). As a result, the PWA has a scheme where a broker only routes queries to index servers serving collections within the interval of interest. Then, the broker merges the results from these index servers and sends them back to the user. Notice that, some web archives use Solr[8] or Lucene instead of NutchWAX [14], but their index partition scheme and architecture are similar.

Queries are also balanced between several replicas of the system with the Linux Virtual Server[9]. In turn, the system replication supports fault tolerance mechanisms and flexibility to adjust the number of replicas according to the expected query throughput. Hadoop is an important piece in scaling out this architecture [29]. Hadoop is a framework that provides distribution, parallelization, load-balancing and fault-tolerance services to software programmed according to the MapReduce model [12]. It is especially well-suited to process large datasets. The PWA currently supports tens of full-text queries per second over a web collection of more than 1.2 billion documents created since 1996. A two tier

[7] `http://archive.pt`
[8] `http://lucene.apache.org/solr/`
[9] `http://www.linuxvirtualserver.org/`

	Wayback Machine	PWA	Everlast
Scalability	High	High	Very High
Reliability	High	High	Medium
Time-awareness	No	Yes	Yes
Performance	High	Very High	Medium

Table 1: Comparison between web archive search architectures' characteristics.

cache mechanism was included to achieve this performance, composed by a search results page cache working in the web server and several caches in the index servers for requested posting lists and metadata constantly used at runtime, such as the documents' timestamps.

4.3 Everlast

P2P architectures have loosely-coupled computers called peers, where each peer operates as both a server and a client aiming to share resources. Everlast is a web archive framework with a peer-to-peer (P2P) architecture for storing and searching past documents [4]. Its indexes are partitioned by term and then by time, as depicted in Figure 2(d). Each term is assigned to a peer responsible for managing the index for that term. In the same way, all versions of a document are stored by a peer.

The power of P2P lies in the capability to provide services with practically unlimited scalability of resources. Some existing P2P systems are formed by millions of peers connected via the Internet [25]. These peers belong to people who participate in the effort to provide a common service, which diminishes drastically the cost of the infrastructure. For instance, peers of Everlast could run within intranets of libraries. However, the peers could be unreliable and transient due to their autonomy, which would result in data loss. This tends to be mitigated by replicating data in several peers. Another problem is that P2P systems are based on decentralized object location and routing schemes, such as Chord [26] or Pastry [23]. The expected number of routing steps to find an object (e.g. document) is $O(log\ n)$, where n is the number of peers in the network. This number of steps and the communication latency via the Internet can degrade the performance leading to response times longer than the users are willing to wait.

4.4 Comparison

Table 1 contrasts the search architectures' characteristics. The Wayback Machine and the PWA distribute their indexes by document. Their main difference is the type of index partition, where the PWA takes the time of collections in consideration and uses it to improve performance (i.e. response time and query throughput).

Everlast enables a higher scalability of storage space and load, but its decentralized coordination over the Internet degrades performance and diminishes reliability. These two factors are quite prohibitive for users who search in web archives as in web search engines and expect the same behavior [11].

Notice that, all system architectures may have index partitions that receive more requests than others. For instance, the PWA may receive more requests on partitions that handle older collections or the Everlast may receive more requests on partitions with some terms. However, all these architectures enable adding more replicas to the overloaded partitions and load-balancing requests among them to overcome this problem.

5. CONCLUSIONS

Finding the desired information in a web archive containing billions of documents acquired throughout time is a challenging task. There are still many technological problems to solve. For instance, the Internet Archive's Wayback Machine, which is the largest web archive in the world, was only able to index all stored data by URL. In this survey, we describe and compare existing web archive search architectures that support time-travel queries. We discuss strategies to partition and distribute indexes by time to speed up and scale out web archive search architectures. All architectures have pros and cons, but we consider that the architectures with indexes distributed by time and then by document, such as the architecture of the Portuguese Web Archive, have additional advantages. These architectures offer a simple way to manage and scale web archive indexes.

We believe that the current web archive search technology can scale out for the already huge amount of historical data held by web archives. However, due to the size of the data there is not yet an economical solution when it comes to provide a full-text search service. Web archives would need computer infrastructures as large as the ones used by commercial web search engines, but since they have much less users, web archives cannot monetize the service to pay for those infrastructures. Economically viable solutions are still needed. We hope this survey may stimulate new and more sophisticated searchable web archives in the future.

6. ACKNOWLEDGMENTS

This work could not be done without the support of FCCN and its Portuguese Web Archive team. We thank FCT for its LASIGE and INESC-ID multi-annual support.

7. REFERENCES

[1] R. Ackland. Virtual Observatory for the Study of Online Networks (VOSON) - progress and plans. In *Proc. of the 1st International Conference on e-Social Science*, 2005.

[2] O. Alonso, M. Gertz, and R. Baeza-Yates. On the value of temporal information in information retrieval. *ACM SIGIR Forum*, 41(2):35–41, 2007.

[3] A. Anand, S. Bedathur, K. Berberich, and R. Schenkel. Index maintenance for time-travel text search. In *Proc. of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 235–244, 2012.

[4] A. Anand, S. Bedathur, K. Berberich, R. Schenkel, and C. Tryfonopoulos. EverLast: a distributed architecture for preserving the web. In *Proc. of the*

2009 Joint International Conference on Digital Libraries, pages 331–340, 2009.

[5] W. Arms, D. Huttenlocher, J. Kleinberg, M. Macy, and D. Strang. From Wayback machine to Yesternet: new opportunities for social science. In *Proc. of the 2nd International Conference on e-Social Science*, 2006.

[6] W. Y. Arms, S. Aya, P. Dmitriev, B. Kot, R. Mitchell, and L. Walle. A research library based on the historical collections of the Internet Archive. *D-Lib Magazine*, 12(2), 2006.

[7] R. Baeza-Yate and B. Ribeiro-Neto. *Modern information retrieval: the concepts and technology behind search*. Addison-Wesley Professional, 2011.

[8] L. A. Barroso, J. Dean, and U. Hölzle. Web search for a planet: the Google cluster architecture. *IEEE Micro Magazine*, pages 22–28, 2003.

[9] K. Berberich, S. Bedathur, T. Neumann, and G. Weikum. A time machine for text search. In *Proc. of the 30th SIGIR Conference on Research and Development in Information Retrieval*, 2007.

[10] M. Burner and B. Kahle. The Archive File Format. `http://www.archive.org/web/researcher/ArcFileFormat.php`, September 1996.

[11] M. Costa and M. J. Silva. Characterizing search behavior in web archives. In *Proc. of the 1st International Temporal Web Analytics Workshop*, 2011.

[12] J. Dean and S. Ghemawat. MapReduce: Simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[13] I. M. Foundation. Web archiving in Europe. Technical report, CommerceNet Labs, 2010.

[14] D. Gomes, J. Miranda, and M. Costa. A survey on web archiving initiatives. In *Proc. of the International Conference on Theory and Practice of Digital Libraries*, 2011.

[15] D. Gomes, A. Nogueira, J. Miranda, and M. Costa. Introducing the Portuguese web archive initiative. In *Proc. of the 8th International Web Archiving Workshop*, 2008.

[16] E. Hatcher and O. Gospodnetic. *Lucene in Action*. Manning Publications Co., 2004.

[17] E. Jaffe and S. Kirkpatrick. Architecture of the Internet Archive. In *Proc. of SYSTOR 2009: The Israeli Experimental Systems Conference*, pages 1–10, 2009.

[18] J. Masanès. *Web Archiving*. Springer-Verlag New York Inc., 2006.

[19] J. Masanès. LiWA news #3: Living web archives. `http://liwa-project.eu/images/videos/Liwa_Newsletter-3.pdf`, March 2011.

[20] J. Michel, Y. Shen, A. Aiden, A. Veres, M. Gray, J. Pickett, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant, et al. Quantitative analysis of culture using millions of digitized books. *Science*, 331(6014):176, 2011.

[21] A. Ntoulas, J. Cho, and C. Olston. What's new on the web?: the evolution of the web from a search engine perspective. In *Proc. of the 13th International Conference on World Wide Web*, pages 1–12, 2004.

[22] M. Ras and S. van Bussel. Web archiving user survey. Technical report, National Library of the Netherlands (Koninklijke Bibliotheek), 2007.

[23] A. I. T. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Proc. of the IFIP/ACM International Conference on Distributed Systems Platforms*, pages 329–350, 2001.

[24] M. Stack. Full text searching of web archive collections. In *Proc. of the 5th International Web Archiving Workshop*, 2005.

[25] R. Steinmetz. *Peer-to-peer systems and applications*, volume 3485. Springer-Verlag New York Inc., 2005.

[26] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proc. of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 149–160, 2001.

[27] B. Tofel. 'Wayback' for Accessing Web Archives. In *Proc. of the 7th International Web Archiving Workshop*, 2007.

[28] G. Weikum, N. Ntarmos, M. Spaniol, P. Triantafillou, A. A. Benczur, S. Kirkpatrick, P. Rigaux, and M. Williamson. Longitudinal analytics on web archive data: It's about time! In *Proc. of the 5th Conference on Innovative Data Systems Research*, pages 199–202, 2011.

[29] T. White. *Hadoop: The Definitive Guide*. Yahoo Press, 2010.

[30] J. Zobel and A. Moffat. Inverted files for text search engines. *ACM Computing Surveys (CSUR)*, 38(2):6, 2006.