Creating a searchable web archive (Technical Report)

Daniel Gomes, David Cruz, João Miranda, Miguel Costa, Simão Fontes
Foundation for National Scientific Computing
Av. Brasil, 101
1700-066 Lisboa, Portugal
{daniel.gomes, david.cruz, joao.miranda, miguel.costa, simao.fontes}@fccn.pt

ABSTRACT

The web became a mass means of publication that has been replacing printed media. However, its information is extremely ephemeral. Currently, most of the information available on the web is less than 1 year old. There are several initiatives worldwide that struggle to archive information from the web before it vanishes. However, search mechanisms to access this information are still limited and do not satisfy their users that demand performance similar to live-web search engines.

This paper presents some of the work developed to create an efficient and effective searchable web archive service, from data acquisition to user interface design. The results of research were applied in practice to create the Portuguese Web Archive that is publicly available since January 2010. It supports full-text search over 1 billion contents archived from 1996 to 2010. The developed software is available as an open source project.

1. INTRODUCTION

Human knowledge has been incrementally built for thousands of years. The new generations augment knowledge transmitted by the previous ones. Inventions such as writing, press and recently the web, deeply improved this process. However, after a short period of time, the information published on the web becomes unavailable and commonly is lost forever. Ntoulas et al. estimated that only 20% of the pages available today will still be available one year from now [34]. Besides loosing important scientific and historical information, web transience causes that common people are loosing their memories as individuals. Everyday, people take photos and share them directly and exclusively on the web without having the most elementary preservation concerns. As consequence, in the future many people will have difficulties in showing portraits of their ancestors or memories.

The web lacks preservation mechanisms. For centuries, organizations such as archives and libraries, ensured the preservation of information published on printed media, for future generations. Since 1996, several web archiving initiatives were created worldwide [17]. Web archives acquire, store, preserve and provide access to information published on the web across time, which also includes contents created before the digital era, that were digitized and published online. These contents include official documents, such as those kept in libraries or museums, but also, commercials, games or pictures that are valuable descriptions of recent history.

Archiving data from the web and preserving it is not enough to make web archives useful for societies. Historical information must be searchable and web users expect a performance similar to the one provided by live-web search engines [37]. However, achieving this goal raises new challenges and search engine technology cannot be directly applied to web archives. Web archives and search engines are complementary. Search engines process online contents hosted on their original servers. There is no concern with content preservation across time. The information they gathere from the web is meant to be permanently updated to be as fresh as possible. On the other hand, web archives address offline contents, frequently in obsolete formats, that must be preserved and reproduced as close as possible to their original layout. Searching web archives has always a temporal dimension that must be addressed on queries and results.

The Portuguese Web Archive (PWA) project began in 2008 and it aims to preserve web contents of interest to the Portuguese community. It was based on the Internet Archive archive-access project tools [22], which are used by most web archives worldwide [17]. However, we observed that these tools did not fulfill our requirements at several levels, from data integration to user interface design. Thus, we researched and developed a new web archive search engine. In January 2010, we released a beta version of a search service over the PWA (available at `http://archive.pt`). In October 2011, the service provided public access to 1 billion contents (10^9). This paper presents the main lessons learned while developing our service, the research that sustained the adopted design decisions and the experience obtained from operating the system in a production environment. Its main contributions are an analysis of the application of deduplication mechanisms during data acquisition from the live web, ranking tunning to support search over historical web collections and a new user interface to support search over web archives. The developed software is available as an open source project (available at `http://code.google.com/p/pwa-technologies/`).

2. RELATED WORK

Ideally, web archives should acquire and preserve every content published on the web. In this sense, web archives are similar to traditional printed media archives and selection policies must be applied to acquire the most relevant information and in an amount of data that can be addressed by the available resources [16, 28]. Capturing web documents to be later reproduced is challenging, because it is necessary to interact with millions of web servers, beyond

our control [10, 35].

Web archives face many challenges related to scalability and information overload because they accumulate previous documents and indexes, unlike web search engines that drop the old versions when new ones are discovered [2]. Web archives already hold more than 181 billion contents and this number continues to grow as new initiatives continue to arise [17]. This data dimension is one order of magnitude larger than the number of documents indexed by the largest web search engine and 150 times more than the content of the Library of Congress. About 89% of the world web archives provide URL search [17], mostly supported by the open source Wayback Machine [24, 43], which returns a list of chronologically ordered versions of that URL. However, this type of search forces the users to know the URLs of the content that contain the required information, some of which may have disappeared many years before.

Costa and Silva studied the information needs and search behaviors of web archive users [13, 14]. The main conclusions are that users from web archives and web search engines have different information needs. However, they maintain the same search behavior when using both types of systems, ignoring that they are searching historical data sets. The National Library of the Netherlands conducted an usability test on the searching functionalities of its web archive [37] and derived a list of the top 10 functions that users would like to see implemented. Full-text search was the first ranked, followed by URL search. At least 67% of web archives support full-text search for a part of their collections [17]. The large majority of full-text search that these web archives support is based on the Lucene search engine [20], which is the core of NutchWAX. However, its performance is considered unsatisfactory by the stakeholders [17, 41]. Cohen et al. showed that the out-of-the-box Lucene produces low quality results, presenting half of the precision of the best systems participating in the TREC Terabyte track [11]. In addition, many of the specific characteristics of web archive collections are not handled by Lucene, degrading the quality of results even more. For instance, Lucene does not contemplate any temporal attribute, such as the crawl date or last-modified date of documents, despite studies showing that users prefer the oldest documents over the newest [13, 14].

Research on the design of user interfaces to search historical web collections is giving its first steps. It has mainly being focused on the exploitation of the temporal perspective of data through the introduction of new user interface elements such as timelines or information clusters [1, 30]. However, the presented research does not address the specific requirements of searching historical web collections. At most, the original data for this research was obtained from curated online news archives, which is not representative of the heterogeneity of data addressed by a web archive. Hearst's book presents a comprehensive analysis of user interface design to search the live web but searching over historical web collections is not addressed [21].

Web archiving research projects have been receiving a growing support from the European Commission. The Living Web Archives (LiWA) was the first project to be funded by the European Commission [29]. LiWA aimed to provide contributions to make archived information acessible and not just stored. It addressed problems shared with other areas such as web spam detection, terminology evolution, capture of stream video, and assuring temporal coherence of archived contents. LiWA was followed by the Longitudinal Analytics of Web Archive (LAWA) which aims to build an experimental testbed for large-scale data analytics [44]. Its focus is on developing a sustainable infrastructure, scalable methods, and easily usable software tools for aggregating, querying, and analysing heterogeneous data at Internet scale. Particular emphasis is given to longitudinal data analysis for web data that has been crawled over extended time periods. The most recently funded project related to web archiving is named ARCOMEM (From Collect-All Archives to Community Memories) that aims to exploit Social Web and the wisdom of crowds to make web archiving a more selective and meaning-based process [4].

3. ACQUIRING WEB DATA

Web archives must acquire content before it disappears. Data adquisition is typically made through web crawling or integration of historical web collections supplied by third-parties.

3.1 Crawling the live web

Crawling is a data acquisition method widely used by web archives that capture information for preservation and later access. Besides the textual contents to support search, web archives must exhaustively gather embedded files to enable the reproduction of the archived pages. Thus, crawling for archiving imposes an higher workload on crawlers and visited web servers in comparison to search engines.

The objective of the PWA is to preserve web contents relevant to the Portuguese community. This relevance criteria is highly subjective and must be transposed to machine-understandable rules. On the other hand, the crawler must be robust to situations harmful to its performance and polite to the visited web servers. The Heritrix crawler v.1.14.3 [32] was used to acquire contents from the live web. It was configured based on the information derived from web characterizations [31] and crawling experiences [19].

The PWA performs two types of crawls from the live web. *Trimestral broad crawls* are performed every 3 months and include a broad set of approximately 500 000 seed URLs derived from the national top-level domain listings, user submissions, web directories of Portuguese speaking countries and home pages of sites successfully harvested on the previous crawl. On average, 78 million contents are downloaded on each crawl (5.9 TB). Its purpose is to archive exhaustive snapshots of the national web. The *daily selective crawls* are performed every day and include a set of 359 online publications selected in collaboration with the National Library, typically online newspapers and magazines. As these types of publications receive heavier load during work hours, our crawl begins at 16:00 and reaches 90% of the URLs to visit at 7:00 of the next day. On average, 764 000 contents are downloaded on each day (42 GB).

The crawler got trapped several times in infinite sites, while performing trimestral broad crawls. There are millions of sites available on the web but they tend to be supported by a small number of publishing platforms. This fact enabled the automatic detection of infinite sites through regular expression matching applied to URLs. We also avoid following links that originate data insertion such as comments or buy actions on well known platforms. The black list and exclusion rules are available at `http://arquivo.pt/`

crawlfilters to be reused and validated by the community.

During our crawls we respect the access rules established by the authors through the Robots Exclusion Protocol (REP). This best practice avoids unwanted access complaints by site owners and avoids harvesting infinite parts of a site such as online calendars. However, it raised some unexpected problems. Social network platforms, such as Facebook are commonly used by organizations and individuals to publish information and make it available in replacement of traditional sites. Several Facebook pages publicly available on the web contain valuable content for preservation. The problem is that by default Facebook is very restrictive regarding crawlers and these rules cannot be changed by users. A formal authorization request to Facebook is required to crawl its contents[1]. Nonetheless, the crawl of some contents like photos is always forbidden. Even with the obtained authorization Facebook pages are difficult to crawl because they are strongly based on AJAX technology and chained redirects, which makes it difficult to discover and adquire embedded or linked contents. Although publicly available Facebook pages may seem equivalent to traditional sites they raise new barriers to web archiving.

Some popular Content Management Systems such as Joomla [8] also present default REP restrictions that do not allow crawlers to harvest all the files required to later reproduce pages. As search engines just need to crawl textual contents to present results from a site, the default REP rules forbid the crawl of embedded files (e.g. CSS, JavaScript or images files) that are mandatory to enable the archive of the complete web page. This situation had a significant impact on the daily crawls. We contacted through email or contact forms the webmasters of the sites that had REP rules to raise awareness about the situation. Only 10% of the sent messages originated a reaction by the webmasters. These results show that default access rules tend to prevail, inhibiting effective preservation of web contents.

3.2 Incremental crawling and deduplication

The amount of data archived is incremental and the waste of resources caused by the storage of duplicates originated by contents that remain unchanged across time is significant. In the contents downloaded in a daily crawl, 46% were duplicated. In trimestral crawls the level of duplication was 30%.

To detect and avoid the storage of duplicates we adopted the DeDuplicator plug-in for Heritrix [39] that analyses the log of the previous crawl and builds an index containing URLs and corresponding cryptographic digests. Heritrix downloads a content from a URL and compares it with the version harvested on the previous crawl. If it remains unchanged, it is discarded. To measure the impact of adding the the DeDuplicator plug-in, we measured the download rate for bouth types of crawl, daily and trimestral. In the daily crawl the download rate was reduced by 2.5%. For trimestral crawls it increased by 0.35%. The obtained results show that the addition of the DeDuplicator did not have a significant impact on the crawler performance.

We ran an experiment to evaluate the storage savings provided by using DeDuplicator. We set a crawl with the original configurations (without DeDuplicator) and another one on the next day using DeDuplicator. The same configurations and seed list were used on both crawls. The obtained results showed that with DeDuplicator 45.6% of the number of crawled contents were not stored because they were marked as duplicates, these documents corresponded to 51.4% of the volume of uncompressed data downloaded during this crawl. Eliminating duplicates added more savings, since the data after being downloaded is compressed. The disk space required was decreased by 76% (from 32 GB to 7.8 GB per day).

We analyzed the distribution of contents stored per media type on both crawls and observed that the prevalence of images dropped from 32.5% on the original crawl to 14% on the deduplicated crawl, while the prevalence of HTML pages increased from 60.1% to 78.8%. This means that images tend to be more persistent and originate more duplicates. These images were published on the web using compressed formats and prevalent textual contents, such as HTML, were not. Thus, the deduplicated collection presented a higher prevalence of textual contents which could be further compressed to save disk space. The original collection presented a compression ratio of 0.70 (46 GB compressed to 32 GB), while the deduplicated collection achieved a compression ratio of 0.36 (20 GB compressed to 7.2 GB). The results also showed that 45.6% of the contents persisted on the web after 24 hours. This result is close to the estimation of 54.6% after 1 day derived from the persistence model proposed by Gomes and Silva [18].

After deploying the deduplicator in a production environment, we compared 60 daily crawls performed without the DeDuplicator (average size per crawl of 27.6 GB of data) against 76 crawls performed using the DeDuplicator (average size per crawl 10.3 GB of data). The obtained results showed that the DeDuplicator enabled an average disk space saving of 62%. The trimestral broad crawls are made with a longer interval between them, which reduces the probability of duplicates occurrence. The obtained results from a comparison of 3 original crawls with 3 deduplicated trimestral crawls showed that on average 28.3% duplicated contents were detected on each crawl, which enabled a storage space reduction of 41.1% per crawl.

The approach adopted by the DeDuplicator does not enable saving bandwidth because the contents must be downloaded for comparison. However, it saves expensive disk I/Os during the crawl by not writing duplicates [19]. Eliminating duplicates in web archives saves a considerable amount of disk space and also reduces the amount of data to be indexed. The average size of the indexes structures that support search over the trimestral broad crawls was reduced by 22.6%, from 1.81 bytes on the index per each indexed URL to 1.4 bytes/URL. Regarding the daily crawls, the index structures were reduced by 19.7% (1.36 to 1.09 bytes/URL). Although there is a larger prevalence of duplicates identified on the daily crawls, as most of them are images that do not contain indexable text, the savings on the index size are similar to the broad crawls.

The elimination of duplicates may be dangerous from a preservation perspective. For instance, if a frequently updated page includes an image that remains unchanged across time, the image is crawled and stored just once, although several new versions of the page are archived. Thus, if the image is lost, all the versions of the page would be presented incompletely. To mitigate this problem, the first crawls of each year are performed without using the DeDuplicator.

[1] https://www.facebook.com/apps/site_scraping_tos_terms.php

3.3 Integration of delivered collections

Web archives integrate past contents that are no longer available online and are delivered for preservation by third-parties. However, these web data reach web archives in heterogeneous formats, media support (e.g. CD-ROMs, backup tapes, original source code) and with scarce associated metadata (e.g. original site URL, publication dates). Making this data searchable implies converting it to a uniform archive format so that it can be automatically processed, such as ARC [9] or its successor standard WARC [23]. WARC is the official standard but ARC is still most widely supported by web archiving tools. Historical web collections obtained from the Internet Archive were delivered in the ARC format to be integrated in the PWA. However, several other web collections were delivered in distinct formats. We had to create specific integration modules to convert each one of these collections to the ARC format.

A common situation is the integration of site backups made on local file systems with unknown or obsolete software. We believe that the large majority of the web collections created by organizations and individuals have been generated using software that was not designed with long-term preservation concerns, such as offline-browsers or through the Save feature of common web browsers. Offline browsers typically do not store meta-data related to each content saved locally, such as the original URL. As consequence, web archives cannot support URL search over these contents. If full-text is supported by the web archive, the contents could still be searchable but link-based algorithms could not be directly applied. Thus, these type of integrations require reverse engineering to model the archive file format and extract content meta-data. For instance, in 1995 a CD-ROM containing a snapshot of the Portuguese web was published as an attachment of a book. The web collection had the original URLs embedded as a reminder within each HTML page. However, non-HTML contents such as images did not have any URL associated. The extraction of the original URLs for these contents was automatized because each site was stored on a different directory and the original URLs were inferred by following relative links from pages. If the page with the URL `http://site.net/index.html` refered the image located in `./0.jpg`, then the original URL for the image was `http://site.net/0.jpg`.

The integration software was developed modularly so that it can partially applied and combined to address recurrent problems in independent collections. The converter of the CD-ROM collection to ARC format is available as an open source project at `http://code.google.com/p/roteiro2arc/`. The software to convert HTTrack crawls to ARC files is available at `http://code.google.com/p/httrack2arc/`. HTTrack is a crawler used by web archives that stores content in a specific format [17]. The main purpose of HTTrack is to create site backups and the web collections generated with it contain most of the meta-data required to be successfully converted to the ARC format.

4. SEARCHING THE PAST WEB

Millions of contents archived across time can match a query from the user. Hence, ranking the contents is essential to provide relevant search results. The ranking of web search results has been thoroughly studied in Information Retrieval. However, searching collections composed by contents harvested form the web along time raises new challenges.

4.1 Historical web graph computation

Link-based ranking algorithms assume that a link created by an author to a URL attributes importance to it. Therefore, the number of links received by a URL is an indicative of its importance. The anchors associated to links provide a short description for the linked content. These descriptions have an additional value because they were created by content readers. Link-based algorithms have been proven to be efficient to help ranking web search results through the analysis of links and anchor texts of the web graph. Therefore, applying these kind of algorithms to historical web collections to improve search results sounds promising. However, link-based algorithms cannot be applied directly to web archives. Each node in a live-web graph is uniquely identified by a URL, but in a web archive each URL is associated to multiple content versions acquired along time (see Figure 1(a)). Thus, the nodes of the graph derived from a historical web collection must be identified by content versions (URL + timestamp) and not just by URLs. Computing the web graph ignoring this fact leads to the following problems:

Links to the future: Figure 1(b) shows that $URLx$ in 1996 links to a version of $URLy$ in 2004. The version of $URLy$ in 2004 can have a completely different content than when the author of $URLx$ created the link;

Link overweighting: Figure 1(c) shows that $URLy$ is overweighted, because it persisted for a long time and not because it received many links from different authors.

The adopted solution was to pre-process the web graph to generate temporal layers between the URL versions before applying link-based algorithms. First, we identify each content's version by the pair (URL, timestamp). We used the timestamp of content acquisition, but other timestamps could also be used, such as the creation or publication date. For each link found on the graph from a $URLx$ to $URLy$, we linked each version of $URLx$ to the closest archived version of $URLy$ within a time window. The time window size varies according to the timestamp of the referrer URL. The time window is defined as $]-\infty, timestamp + t[$, where t varies according to the timespan of the collection. For instance, a collection integrated from the Internet Archive was composed by contents from domain .PT between 1996 and 2000. We set t to one month because the distribution of the crawl dates suggested that the collection was generated from monthly web crawls. For the daily crawls, we set t to one day. The time window limitation eliminates links to versions in a far future. Limiting links to the closest version of the referenced URL eliminates the problem of link overweighting. The proposed pre-processement algorithm also reduces the number of links in the web graph.

4.2 Ranking model

Supervised learning algorithms have been employed to tune the weights between combined ranking features resulting in significant improvements [27]. Learning algorithms require specific Information Retrieval test collections. However, existing test collections do not address web archive requirements [12]. For instance, their data sets are not composed of historical data and the query sets have no temporal

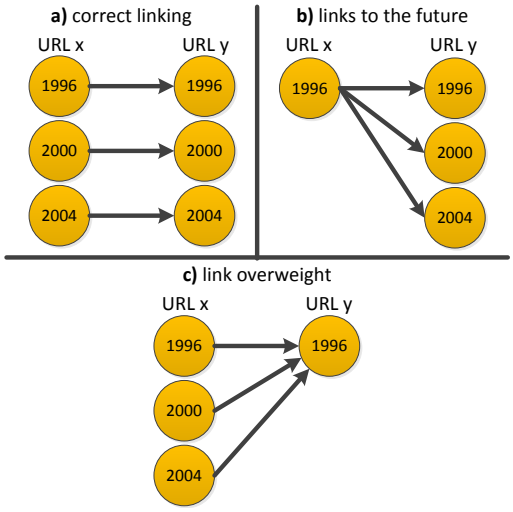


Figure 1: Problems with the computation of link graphs generated from historical web collections.

needs. Therefore, existing test collections are not directly applicable to web archives.

Without access to historical web collections we derived our ranking model from the TREC collections. We used the TD2003 and TD2004 datasets from the TREC 2003 and TREC 2004 web tracks, that included the .gov collection.

For each query, and using Okapi BM25[38] for a base line, we extracted 1 000 top contents. The resulting dataset was scored by 30 ranking features. We used a ranking feature selection algorithm to remove irrelevant and redundant features. Four features were selected from the 30, and those were: Lucene, $\text{MinSpanCov}_{\text{unord}}$, $\text{MinSpanCov}_{\text{unord}}$ and $\text{MinSpanCov}_{\text{ord}}$. The selection algorithm selects in each iteration, the feature f_i that leads to the highest gain when combined with the previously selected features S. The algorithm iterates until the gain of adding a new feature f_i is lower than a defined threshold. The weights among the subset of features $S + f_i$ are tuned in each iteration by the learning to rank algorithm (L2R) called SVM-MAP[27, 45]. Thus, the computational cost of processing many ranking features was balanced against the ranking quality.

The produced ranking model is composed by the linear combination of the four weighted functions presented in Table 1. The scores from the n selected ranking features are added after each feature f_i is weighted by a coefficient λ_i. For a content d with a vector of low-level ranking features associated, $\vec{d}$:

$$\text{rankingModel}(\vec{d}) = \sum_{i=1}^{n} \lambda_i f_i(\vec{d}) \quad (1)$$

Lucene was the only term weighting function selected, being preferred to BM25 and TFxIDF. Adding the latter also did not improve significantly the quality of results. On the other hand, adding functions that quantify the distance between query terms did improve. The assumption is that documents with smaller distances between query terms should have a higher score, since these terms have a higher chance of being related. The selected ranking model includes the

function f_i	weight λ_i
Lucene	0.02332
$\text{MinSpanCov}_{unord}$ - title	0.59394
$\text{MinSpanCov}_{unord}$ - content	0.34503
MinSpanCov_{ord} - anchor	1.25928

Table 1: Features selected for the Portuguese Web Archive ranking model.

$\text{MinSpanCov}_{unord}$ function over the title and content fields. The function returns the length of the shortest segment of text between the two terms. This length is then transformed by an exponential decay function based on [42]:

$$\text{MinSpanCov}(q, d, span) = \ln(1 + e^{-span-length(q)+1}) \quad (2)$$

A variant of this function, denoted MinSpanCov_{ord}, was selected over the anchor field. In this, the shortest segment of text must cover each term in the same order as in the query.

The presented methodology will be applied over a test collection that has been created especially for web archives. Then, ranking features based on temporal information will also be experimented.

5. USER INTERFACE DESIGN

Most web archives were created and are maintained by libraries [17]. The digital library user interfaces aim to provide several search choices to the users through a stricter model of interaction based of faceted search (e.g. title, author, abstract). This leads to complex user interfaces composed by several UI elements that require strong contextualization and decisions by the users to provide relevant search results. On its turn, the typical search engine interface is simpler and more familiar to users [15] which diminishes the learning curve. The downside is that it usually does not consider the temporal dimension.

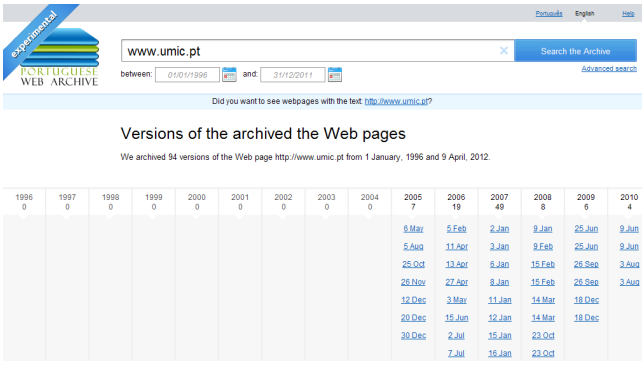


Figure 2: Result page for a URL search on the Portuguese Web Archive (history page).

The choice of the search UI for the PWA was conditioned by the adopted data acquisition policy. Digital libraries host carefully curated collections with rich curated meta-data to enable faceted search interfaces, but the PWA broadly archives large amounts of web data as search engines do.

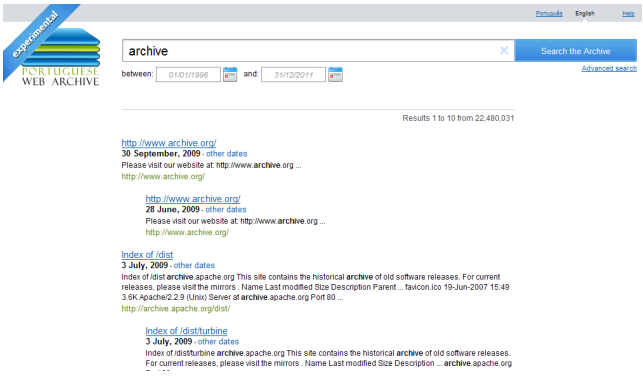
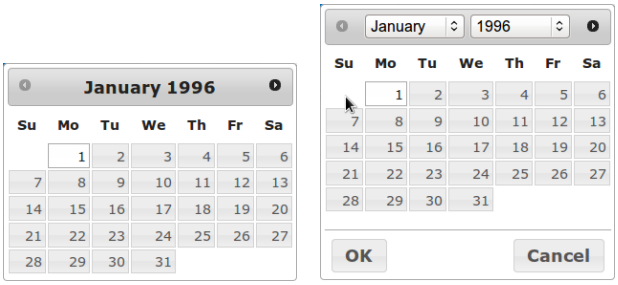


Figure 3: Result page for a full-text search on the Portuguese Web Archive.



(a) (b)

Figure 4: Changes in the datepicker: (a) JQuery's datepicker original layout (b) Datepicker adapted to web archive interface.

Thus, we decided to use a typical web search engine interface as baseline. A web archive UI must additionally address temporal search restrictions (e.g. definition of date interval), versioning of URLs on search results (e.g. version comparison) and reproduction of archived contents with meta-data for temporal contextualization (e.g. present crawl date).

During the development of our UI we performed several iterations of laboratory usability tests to identify interaction problems and validate changes. Summarily, each testing round consisted of 10 tasks presented to 6 users that performed them using the "think aloud method" [33]. Each of the users executed the test individually in the presence of an usability expert. The audio and screen captures of the user sessions were recorded for later analysis. Each user filled a pre-questionnaire to establish a profile and a post-questionnaire for measuring their satisfaction [26].

The PWA interface is available at `archive.pt` and it is composed by:

Archived content view: presents the archived content along with the original URL and crawl date. It supports link navigation within the archive;

URL search results list (Figure 2): presents the history of crawled versions from a given URL in a yearly grid. Each date links to the archived content view;

Full-text search results list (Figure 3): for each search result presents the title that links to the archived content, its crawl date, a *view history* link to the history page of that URL and a snippet of the content containing the query terms;

Search form: the search form is present on the top of the URL and full-text lists. It is composed by a text search box that receives the query terms and two datepickers to restrict the crawl dates of the contents to be searched;

Advanced search form: enables users to refine search by defining phrase, term negation, results sorting, crawl dates, file format, site restriction and number of presented results.

The users compared the behavior of their favorite search engine with our web archive and expected the same response speed and search results quality due to the UI similarity. They did not understand the difference between searching the live web and historical web collections. Using a web archive to access pages that are no longer available on the live web is a confusing concept to most users and requires technical knowledge about the functioning of the Internet. Only 20% of the users answered that they knew what a web archive was. They typically ignored the presented dates and links to the history on the full-text search results list. However, when they were conducted to the history page (URL search results list), the grid layout enabled them to perceive the versions' distribution among the years. The original Wayback Machine interface used the symbol "*" to mark new versions of a URL, but the meaning of this symbol was confusing to the users. As we acquire data using the DeDuplicator, duplicates are not presented on the history page. Thus, we removed the "*" symbol.

Initially, the date presented on the full-text search results list and archived content view was labeled as "crawl date". However, the technical meaning of "crawl" was not understandable to the users. After several tries, we found that the best option was to omit any label and let the users interpret the meaning of the presented date in the way that would be easier for them to understand it (e.g. original date, crawl date, archived date).

Throughout our usability testings we observed that users did not have difficulties with the search form. It consists of elements that users are used to find in search engines, a text field and a submit button, but also two date-picker for delimiting the timespan of the search. However, the introduction of the datepickers raised unexpected challenges because the adoption of a conventional datepicker, obtained from the jQuery UI library (Figure 4(a)), did not meet the web archive users expectations. Conventional datepickers are meant to be used to specify days or short intervals of time, but in web archives the time intervals can be very small, such as a specific day, or very broad, spanning several years. The datepicker became a problematic UI element that required several design and evaluation iterations.

Figure 4(b) presents the final version of the datepicker. The datepickers must be complemented with text fields to enable direct typing of dates. The typical layout of datepickers with left/right arrows is useful for month navigation but did not work to define date ranges of several years (see Figure 4(a)). Thus, dropdown lists for month and year selection were added.

Conventional datepickers were designed assuming that users select a date in the following order: year, month, day. Therefore, when users click on a day, it means that the selection is finished and the datepicker closes. The obtained results showed that 10% of the users repeatedly clicked on the day, which caused the datepicker to immediately close, before understanding the problem and working it around. Moreover, we observed that users tried to select only the month and year but they did not want to selected a specific day. Their mental focus was on the temporal granularity of months, so choosing a specific day did not make sense to them. We changed the conventional datepicker behavior to use default values for day selections. The first day of the month is the default for the starting date and the last day of the month for the end date. Some users chose the month and year and then clicked outside the datepicker to close it. They expected the datepicker to save their selections. Thus, the closing behavior is not equally perceived the same way by the users. For some, clicking outside the datepicker closes it, saving the chosen date. For others, clicking outside cancels the changes. We solved this problem by having "OK" and "Cancel" buttons at the bottom of the datepicker.

The PWA supports full-text and URL search. Our first UI versions were composed by two distinct search forms: one for full-text search and another one for URL search. This approach failed because users did not understand the difference between search types. They inserted full-text queries on the URL search form, and vice-versa. The fact may be justified by the users' tendency to fill the first text field that looks like a searchbox [33]. The solution was to present a single textfield that receives any query. If the query term is composed by a URL, the corresponding history page is presented to the user. If the query terms include a URL but also other terms, it does a full-text search with the query terms but also presents a suggestion link to the history page of the queried URL. Otherwise, it performs a full-text search for the query terms. The URL queries are expanded to find results crawled with different URLs, that are likely to present reference the same content, for example, with and without "www." prefix, trailing "/" or "index.html" string.

The addition of a query spellchecker had great impact on the perceived quality of the web archive. Users frequently mistyped queries and blamed the web archive for poor search results, often failing to spot their own mistypes. After the introduction of the spellchecker, there were less negative user comments. The adopted spellchecker is based on Hunspell.

The presented changes increased the overall user satisfaction from 51% on the first version of the UI to 71% on the last one.

There are still many challenges to be tackled, such as improved aesthetics, clearer temporal information or better archived page presentation.

6. SEARCH PERFORMANCE RESULTS

This Section presents preliminary results regarding search speed and effectiveness. Further research with larger setups and deeper analysis is required. However, we could not find related work to compare the obtained results. We believe that this is the first time that a performance evaluation of a search system over a web archive is published.

6.1 Response times

The performance of the PWA search system was experimentally measured following a blackbox approach. The web collection was composed by 147 million contents gathered from 1996 to 2007. The experimental setup was composed by one load balancer that distributed the queries among 7 replicated search servers. The load balancing mechanism was implemented using the Linux Virtual Cluster software (Linux Server Cluster for Load Balancing). Each search server supported queries over the full collection. The search servers shared the data storage device through a Storage Area Network that held the index. Each machine had 2 Xen Quad-core CPUs, 32 GB of memory and ran Linux. An increasing number of queries were submitted in parallel during a fixed interval of 5 minutes using several instances of the JMeter sofware and it was measured the time taken by the system to respond to each query. The query set used to simulate the workload was composed by 300 000 queries obtained from a Portuguese web search engine [40] because web archive query log data sets were not available. Figure 5 presents the relation between workload and response time supported by the system. The obtained results show that until an average workload of 20 responses per second, the system is able to maintain an average response time of approximately 1 second. However, when the workload reaches 25 responses per second, the average response time increases to 1.5 seconds and the system reaches its exhaustion point. From this point, we continued to increase the number of queries issued to the system but it was unable to respond to them. Thus, the system enters a thrashing state caused by overload.

The experimental setup previously described was deployed to production and we analyzed the logs of the queries issued by real users between May 2010 and July 2011 over a web collection of 187 million contents. Table 2 presents

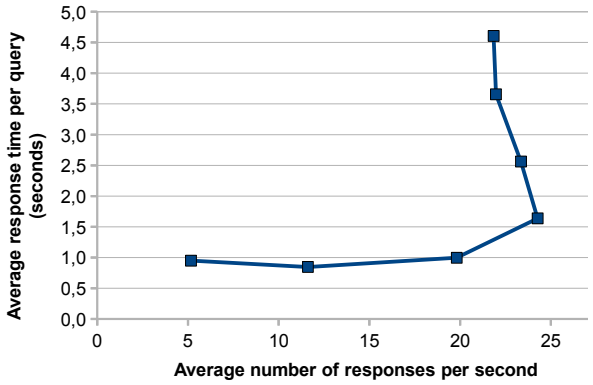


Figure 5: Experimental performance results: relation between average response time and workload.

Response time (s)	%full-text queries	%URL queries
$[0, 1[$	62.9%	71.7%
$[1, 2[$	14.9%	11.7%
$[2, 3[$	9.9%	6.5%
$[3, 4[$	4.5%	1.4%
$[4, 5[$	2.3%	2.2%
$[5, \infty[$	5.5%	6.6%

Table 2: Response time distribution derived from query log analysis (seconds).

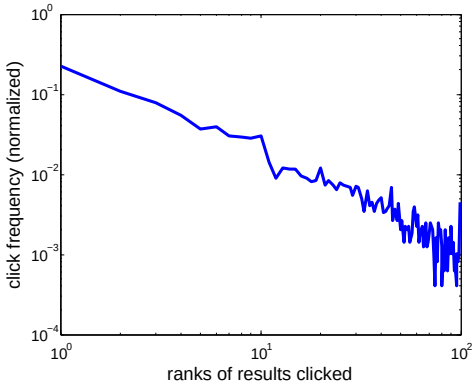


Figure 6: Rank distribution of the clicked results.

the response time distribution for full-text and URL queries. Around 87.7% and 89.9% of the full-text and URL queries, respectively, were responded in less than 3 seconds.

6.2 Results quality

Measuring the quality of web archive search results usage requires standard Information Retrieval metrics. To obtain good results there is a need for adequate test collections, and these are not yet available [12]. Therefore, we measured the quality of our search results by performing a user click-through analysis derived from the query logs of the PWA gathered from June to December, 2010. The obtained results showed that 66% of the clicks were made on the first page of results. Figure 6 shows that the distribution of the ranked position of the results clicked by the users fits a power law distribution with a correlation value of 0.88. For instance, 23% of the clicks were made by the users on the first result presented by the system and 12% on the second one. These results are similar to those presented on web search engine studies [5]. Thus, they are a positive indicator of relevance. Nonetheless, Joachims et al. showed that users tend to indistinguishly click on the results on the top of the lists [25]. They scan results from top to bottom and trust that web search engines present the most relevant results on the first positions. Another positive indicative is that only 2% of the URL search sessions did not receive any click by the users.

On the other hand, we obtained two negative quality indicators. The first one, is that 31% of the full-text sessions did not receive any click by users. This abandonment rate suggests that users quit search before finding what they needed. The second negative indicative is that 85% of users identified by IP address did not revisit the web archive during the seven months period. One possible explanation for the non-revisit figure is that most users do not have a frequent need to search for historical web contents as they do for current information. Hence, the interval of time for users to revisit a web archive tends to be longer than for search engines. A longer time interval between revisits also reduces the probability of the same user revisiting the web archive using the same IP address.

7. CONCLUSIONS AND FUTURE WORK

This study shares the experience obtained while creating a fully searchable web archive from data acquisition to user interface design. We concluded that excluding contents that remain unchanged across time significantly saves storage but has less impact on the full-text index structures because most of the duplicates are images. Web collections are delivered to web archives in multiple heterogeneous archive formats. However, some problems are recurrent and solutions can be reapplied. Well known web-graph based ranking algorithms cannot be directly applied to historical web collections because they assume that each graph node is uniquely identified by a URL and ignore the temporal perspective of data. Web search user interfaces must be adapted to web archives. Users already have a very well defined expectation about search interfaces and they are not receptive to new UI elements, even if they are conventional elements such as a datepicker. However, the usability of web archive search interfaces can be significantly improved by tweaking conventional UI elements.

Web archives have been storing information for years. However, search over historical web collections is giving its first steps. The source code of the developed web archive is available as an open source project (`http://code.google.com/p/pwa-technologies/`). Several questions were raised that require further research, such as the creation of test collections to support Information Retrieval evaluations or the study of alternative user interfaces. Nonetheless, we believe that the provided contributions represent a breakthrough in web archiving and are a baseline to develop more sophisticated searchable web archives in the future. A significant research effort is required to make historical information as accessible as the current web. Achieving this goal would cause that for the first time in the history of mankind, original historical content would be broadly accessible.

8. ACKNOWLEDGMENTS

We acknowledge Marco de Sá and Rui Lopes for their precious collaboration in the user interface design. The Portuguese Web Archive initiative was co-funded by MEC/UMIC and POS_C/EU.

9. REFERENCES

[1] O. Alonso, M. Gertz, and R. Baeza-Yates. Clustering and exploring search results using timeline constructions. In *Proc. of the 18th ACM Conference on Information and Knowledge Management*, pages 97–106, 2009.

[2] O. Alonso, J. Strötgen, R. Baeza-Yates, and M. Gertz. Temporal Information Retrieval: Challenges and Opportunities. In *Proceedings of the 1st International Temporal Web Analytics Workshop (TWAW 2011)*, pages 1–8, 2011.

[3] E. Amitay, D. Carmel, M. Herscovici, R. Lempel, and A. Soffer. Trend detection through temporal link analysis. *American Society for Information Science and Technology*, 55(14):1270–1281, 2004.

[4] ARCOMEM. About arcomem. http://www.arcomem.eu/about/, October 2011.

[5] R. Baeza-Yates, C. Hurtado, M. Mendoza, and G. Dupret. Modeling user search behavior. In *Proc. of the 3rd Latin American Web Congress*, page 242, 2005.

[6] Z. Bar-Yossef, A. Z. Broder, R. Kumar, and A. Tomkins. Sic transit gloria telae: towards an understanding of the web's decay. In *Proc. of the 13th International Conference on World Wide Web*, pages 328–337, 2004.

[7] K. Berberich, M. Vazirgiannis, and G. Weikum. Time-aware authority ranking. *Internet Mathematics*, 2(3):301–332, 2005.

[8] S. Burge. *The Joomla SEO Book* . Alledia Inc., 2007.

[9] M. Burner and B. Kahle. WWW Archive File Format Specification. `http://pages.alexa.com/company/arcformat.html`, September 1996.

[10] C. Castillo. Effective web crawling. In *ACM SIGIR Forum*, volume 39, pages 55–56, 2005.

[11] D. Cohen, E. Amitay, and D. Carmel. Lucene and Juru at Trec 2007: 1-million queries track. In *Proc. of the 16th Text REtrieval Conference*, 2007.

[12] M. Costa and M. J. Silva. Towards information retrieval evaluation over web archives. In S. Geva, J. Kamps, C. Peters, T. S. A. Trotman, and E. Voorhees, editors, *Proceedings of the SIGIR 2009 Workshop on the Future of IR Evaluation*, Boston, Massachusetts, July 2009. IR Publications, Amsterdam.

[13] M. Costa and M. J. Silva. Understanding the information needs of web archive users. In *Proc. of the 10th International Web Archiving Workshop*, pages 9–16, 2010.

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

[15] C. De Rosa, J. Cantrell, J. Hawk, and A. Wilson. *College Students' Perceptions of Libraries and Information Resources: A Report to the OCLC Membership*. OCLC, 2006.

[16] D. Gomes, S. Freitas, and M. J. Silva. Design and selection criteria for a national web archive. In *Proc. of the 10th European Conference on Research and Advanced Technology for Digital Libraries*, pages 196–207, 2006.

[17] D. Gomes, J. Miranda, and M. Costa. A survey on web archiving initiatives. In *International Conference on Theory and Practice of Digital Libraries 2011*, Berlin, Germany, September 2011.

[18] D. Gomes and M. J. Silva. Modelling information persistence on the web. In *ICWE '06: Proceedings of the 6th international conference on Web engineering*, pages 193–200, New York, NY, USA, 2006. ACM Press.

[19] D. Gomes and M. J. Silva. The Viúva Negra crawler: an experience report. *Softw. Pract. Exper.*, 38(2):161–188, 2008.

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

[21] M. Hearst. *Search User Interfaces*. Cambridge University Press, 2009.

[22] Internet Archive. Nutchwax - Home Page. `http://archive-access.sourceforge.net/`, March 2008.

[23] I. ISO. 28500: 2009 Information and documentation-WARC file format, 2009.

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

[25] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proc. of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 154–161, 2005.

[26] J. R. Lewis. Ibm computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. *Int. J. Hum.-Comput. Interact.*, 7:57–78, January 1995.

[27] T. Liu. *Learning to Rank for Information Retrieval*, volume 3 of *Foundations and Trends in Information Retrieval*. Now Publishers Inc., 2009.

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

[29] J. Masanès. Liwa news #1: Living web archives. http://liwa-project.eu/images/publications/LiwaNews1.pdf, January 2009.

[30] M. Matthews, P. Tolchinsky, R. Blanco, J. Atserias, P. Mika, and H. Zaragoza. Searching Through Time in the New York Times. In *Proc. of the 4th Workshop on Human-Computer Interaction and Information Retrieval*, pages 41–44, 2010.

[31] J. Miranda and D. Gomes. An Updated Portrait of the Portuguese Web. In *14th Portuguese Conference on Artificial Intelligence (EPIA 2009)*, Aveiro, Portugal, October 2009.

[32] G. Mohr, M. Kimpton, M. Stack, and I. Ranitovic. Introduction to Heritrix, an archival quality web crawler. In *4th International Web Archiving Workshop (IWAW04)*, Bath, UK, September 2004.

[33] J. Nielsen and H. Loranger. *Prioritizing Web Usability*. New Riders, 2006.

[34] A. Ntoulas, J. Cho, and C. Olston. What's new on the web?: the evolution of the web from a search engine perspective. In *Proceedings of the 13th international conference on World Wide Web*, pages 1–12. ACM Press, 2004.

[35] C. Olston and M. Najork. Web Crawling. *Information Retrieval*, 4(3):175–246, 2010.

[36] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford Database Group, November 1999.

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

[38] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In *Proc. of the Text REtrieval Conference*, pages 109–127, 1995.

[39] K. Sigurdsson. Managing duplicates across sequential crawls. In *6th International Web Archiving Workshop (IWAW06)*, Alicante, Spain, September 2006.

[40] M. J. Silva. The Case for a Portuguese Web Search Engine. In P. Isaias, editor, *Proceedings of IADIS International Conference WWW/Internet 2003*, Algarve, Portugal, November 2003.

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

[42] T. Tao and C. Zhai. An exploration of proximity measures in information retrieval. In *Proc. of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 295–302, 2007.

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

[44] 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 *Proceedings of the 5th Conference on Innovative Data Systems Research*, pages 199–202, Asilomar, California, January 2011.

[45] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *Proc. of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 271–278, 2007.