

# Profiling Web Archival Voids for Memento Routing

Sawood Alam  
Wayback Machine  
Internet Archive  
San Francisco, California, USA  
sawood@archive.org  
0000-0002-8267-3326

Michele C. Weigle  
Department of Computer Science  
Old Dominion University  
Norfolk, Virginia, USA  
mweigle@cs.odu.edu  
0000-0002-2787-7166

Michael L. Nelson  
Department of Computer Science  
Old Dominion University  
Norfolk, Virginia, USA  
mln@cs.odu.edu  
0000-0003-3749-8116

**Abstract**—Prior work on web archive profiling were focused on *Archival Holdings* to describe what is present in an archive. This work defines and explores *Archival Voids* to establish a means to represent portions of *URI* spaces that are not present in a web archive. *Archival Holdings* and *Archival Voids* profiles can work independently or as complements to each other to maximize the *Accuracy* of *Memento Aggregators*. We discuss various sources of truth that can be used to create *Archival Voids* profiles. We use access logs from *Arquivo.pt* to create various *Archival Voids* profiles and analyze them against our *MemGator* access logs for evaluation. We find that we could have avoided more than 8% of additional *False Positives* on top of the 60% *Accuracy* we got from profiling *Archival Holdings* in our prior work, if *Arquivo.pt* were to provide an *Archival Voids* profile based on *URIs* that were requested hundreds of times and never returned any success responses.

**Index Terms**—MementoMap, Memento Routing, Memento, Web Archiving, Archive Profiling, Archival Voids

## I. INTRODUCTION AND MOTIVATION

Web archives capture web pages from the live web, index them, and make them available for replay later. The *Memento* protocol [1] provides a uniform means to lookup archived resources in various web archives in the form of a list of all the captures, or mementos, of a web page or a specific memento closest to a given date and time in the past. The number of public web archives of varying sizes and collection policies supporting this protocol natively or through proxies continues to grow [2], [3], [4], [5].

Discoverability of archived resources is in the interest of both users and web archives. Users want to find mementos that accurately represent the state of the resource at a given time in the past which is more likely to be the case if there are many mementos of the same resource (potentially in many different archives) over the period of the life of the web page. Web archives want their collections to be utilized whenever they have a memento that a user might be interested in. *Memento Aggregators* are services that perform lookup for mementos of web pages across many different web archives using the *Memento* protocol and provide consolidated results. Without an aggregator, small archives will never get the exposure that the *Internet Archive (IA)*<sup>1</sup> gets. A rudimentary implementation of a *Memento Aggregator* that broadcasts lookup requests to every known web archive would be wasteful and inefficient.

By profiling the holdings of web archives an aggregator can perform a more informed routing to only a few potential archives that are likely to return good results for a given *Uniform Resource Identifier (URI)* lookup.

In previous works various means were explored to summarize holdings of web archives to improve the *Accuracy* of lookup routing as discussed in Section II. However, those approaches often resulted in significant *False Positives* when routing lookups. In order to not route lookup requests to the archives that are not likely to return good results, it can be useful to learn about the voids of various archives. In this work we explore various means to identify resources that are not present in an archival collection or that a web archive is not willing to serve. This will further improve the routing *Accuracy* by reducing *False Positives*.

We introduce the term *Archival Voids* to describe what is missing from a web archive as opposed to *Archival Holdings* that describe what a web archive holds. This can be defined as a function that takes two arguments, a *URI Key* and a web archive, and returns the measure of the *Archival voids* under the given *URI* scope (e.g., a top-level domain (*TLD*), a domain, or a domain name with a path prefix) in the archive. However, a reliable estimate of a void requires knowing the set of *URIs* under the scope that ever existed and the set of *URIs* under that scope that are present in the archive. Knowing the cardinalities and overlap or difference of these sets is often not practical as it might require crawling a whole domain or *TLD*. However, in some cases it is possible to estimate the *Archival Voids*. For example, a webmaster who knows all the *URIs* of a website with finite resources can query a web archive to know how many of those resources are present in or absent from the archive. In some cases, it might be possible to estimate the number of *URIs* in scope by querying a public search engine and extracting the number of hits, but this number may not be reliable as it may not contain historical pages or non-textual resources. That said, in this work we are generating an *Archival Voids* profile based on what a web archive knows, so in this case we will only report portions of the web that have zero resources archived/accessible and are requested frequently.

“Why do we care about *Archival Voids*?” This is an obvious question to ask, especially after knowing what is present in an archive. One might argue that if we already know what is

<sup>1</sup><https://web.archive.org/>

present in an archive then everything else can be considered to be missing from the archive. This statement will be true if we had a complete knowledge profile of the archive, which is often not practical and has its own issues when it comes to freshness. On the contrary, if we had an *Archival Holdings* profile based on *URI* sampling then we may not have an accurate knowledge of what is present in the archive, hence we cannot deduce what is not present in it. Similarly, when we have a summarized profile (i.e., using prefixes of *URIs* and not the full *URIs*), we may conclude many *URIs* to be present in an archive, but they might be absent from it (i.e., *False Positives*).

To understand this, assume that an archive holds resources at paths “/a/1”, “/a/2”, “/a/3”, “/b/1”, and “/b/2” under the “example.com” domain. This needs five different keys in the profile to describe these holdings, but we can summarize it as “com,example)/a/\*” and “com,example)/b/\*” (here we are using wildcard character to illustrate that we have all variations at the path depth 2). While this summary ensures that we do not assume “/c/1” is present, it does suggest that “/a/1/z”, “/a/4” and “/b/3” (and many others) are present. If we could list or summarize resources that applications might be interested in, but are not present in the archive, we can further improve the *Accuracy* by reducing *False Positives*.

An aggressively summarized *Archival Holdings* profile improves the freshness, but inherently introduces many *False Positives*. An *Archival Voids* profile can compensate for that by identifying those *False Positives* and explicitly denying their presence in the archive. This means an *Archival Holdings* profile and an *Archival Voids* profile can work together as opposing forces to find the sweet spot for an increased routing *Accuracy* while minimizing the profile size and maximizing freshness.

An *Archival Voids* profile has some use cases beyond *Memento Routing*. For example, an archive can identify voids in its collections to crawl those resources and fill the cavities while those resources are still alive on the web. Another use case could be public disclosure of resources that an archive does not want to collect/serve due to their collection policies. Moreover, it can be helpful in coordinating with other archives, like International Internet Preservation Consortium (IIPC)<sup>2</sup> members do. For example, if an archive has a void in a specific *URI* space, but another archive has holdings for the same, then they have complementary holdings.

## II. RELATED WORK

Query routing is the task of identifying suitable sources of information from a larger set of sources for a given query. It avoids broadcasting, saves resources, and makes the lookup efficient. Query routing is a rigorously researched topic in various fields including networked databases, meta searching, and search aggregation. However, query routing has not been explored in the context of *Memento Routing* extensively. Gravano et al. described protocol and system

called STARTS [6]. The aim of this system was to enable searching across multiple document sources with different interfaces and query models. Liu describes a query routing system to better handle multiple relevant results for a given keyword query [7]. Their system builds profiles of data sources as well as user queries. By combining these two independent profiles a better relevance can be achieved. Callan et al. describe an approach of query-based language model creation for query routing [8]. They reported that running about one hundred queries and retrieving a few hundred documents is sufficient to create a reasonably accurate language model of a textual database for query routing. Lu and Callan described a federated search query routing systems in hierarchical hybrid peer-to-peer networks [9], [10]. Sugiura and Etzioni described the architecture of an automatic query routing system called *Q-Pilot* [11]. They built topic models of 144 specialized search engines to dynamically identify the best subset of candidate search engines for given search queries. Tran and Zhang described a query routing system from structured and linked datasets [12]. Their system returns top-*k* potential data sources or combinations for a given query keyword based on a multi-level scoring mechanism. Moreover, Meng et al. surveyed meta searching techniques [13], Klusch et al. briefly reviewed semantic web service search [14], and Greengrass extensively surveyed information retrieval [15].

Query routing would be a critical component of a *Memento Aggregator* that aggregates a large set of web archives. A *Memento Aggregator* needs to identify a subset of candidate web archives that are likely to return good results for a given lookup *URI*. In traditional information retrieval systems it is easy to route queries when given query terms/phrases have enough signals to identify their membership to certain topics or collections. However, in *URI* lookup routing given *URIs* can be opaque, resulting in lack of signals for classification. For example, the lookup *URI* <https://cdc.gov/coronavirus/2019-ncov/> has sufficient tokens to identify that it may be present in *COVID19*-related web archival collections, but <https://youtube.com/watch?v=QNo5ZDvKuHg> does not<sup>3</sup>. This is why in this work we only rely on the structural features of a *URI*, not the natural language semantics, both for profiling and lookup routing.

Ainsworth et al. attempted to answer the question, “How much of the web is archived?”, in 2011 [16]. Their results showed the answer to this question depends on how we sample the web. For example, they found that the number of *URIs* that had at least one archived copy in any of the web archive were as low as 35% in one sample and as high as 90% in another sample. They reported that about 15% to 31% (depending on the sample) *URIs* are archived at least once per month. Alkwai et al. revisited the archival rate question in 2015, but for web pages of specific languages [17], [18]. They collected over 15,000 *URI* samples from English, Arabic, Danish, and Korean languages to find out how much of the pages from each

<sup>2</sup><https://netpreserve.org/>

<sup>3</sup><https://youtube.com/watch?v=QNo5ZDvKuHg> points to the official “CDC Briefing Room: *COVID-19* Update and Risks” video.

of these languages are archived. They found that 72%, 53%, 36%, and 33% of their sampled *URIs* were archived in these languages, respectively. The GDELT Project, a platform that monitors the world’s news media, reported in 2015 that around 2% of the news articles disappear in a couple of weeks and up to 14% in a couple of months [19]. Similarly, SalahEldeen and Nelson reported that about 11% of the resources shared on social media during the 2011 Egyptian Revolution were lost after a year [20]. This is an alarming rate with which resources on the web disappear. Leetaru investigated how much of the web is being archived by the *Wayback Machine* of *IA* [21]. He noted that we have limited understanding of what is inside of massive web archival datasets, which is one of the core motivations of our work towards web archive profiling. Hallak estimated recently that almost two thirds of the web traffic is not publicly archivable because it goes to sites that are behind session walls or paywalls, to which some social media sites are big contributors of [22]. Kelly et al. developed a framework to archive the private web and integrate it with the public web to fill some of these cavities [23], [24]. These works identify *Archival Voids* as they show some biases in web archiving as well as quantify the small portion of the web many archives hold.

There exist some prior work on archive profiling, including some of our own work. Sanderson et al. created comprehensive content-based profiles [25], [26] of various IIPC member archives by collecting their *CDX* (i.e., *Capture Index*) files and extracting *URI-Rs* (i.e., *Original Resource URIs*) from them. This approach gave them complete knowledge of the holdings in each participating archive, hence they can route queries precisely to archives that have any mementos for the given *URI-R*. This approach yielded no *False Positives* or *False Negatives* (i.e., 100% *Accuracy*) while the *CDX* files were fresh. However, these collected *CDX* files would go stale very quickly because many web archives keep crawling the web constantly and add more mementos to their collections regularly after indexing batches of crawled data. In addition to that, on-demand web archives such as *Save Page Now* (a service from *IA* to submit *URIs* for immediate archiving) [27] add hundreds of mementos to their collections every second [28] and make them available almost immediately. Acquiring fresh *CDX* files from various archives and updating these profiles regularly is not easy. In contrast, AlSum et al. explored a minimal form of archive profiling using only the *TLDs* and *Content-Language* [29], [30]. They created profiles of 15 public archives using access logs of those archives (if available) and fulltext search queries. They found that by sending requests to only the top three archives matching the criteria for the lookup *URI* based on their profile, they can discover about 96% of *TimeMaps* (i.e., the list of mementos of a given resource). This minimal approach had many *False Positives*, but no *False Negatives*. Later, Bornand et al. implemented a usage-based approach for *Memento Routing* by building binary classifiers from *LANL’s Time Travel Aggregator* cache data [31]. They analyzed responses from various archives in the aggregator’s cache over a period of time to learn about the holdings of different archives. They reported a 77% reduction

in the number of requests and a 42% reduction in response time while maintaining 85% *Recall*. Klein et al. revisited the performance of the above binary classifier-based approach after running the service for over a couple of years [32]. They reported an average *Recall* of about 0.73 (i.e., about 12% reduction from the originally reported results) which means the classifier misses more than one quarter of resources that are present in a given archive. They also reported that a more frequent retraining of the models can improve the prediction *Accuracy*. In previous work [33], [34] we explored the middle ground where archive profiles are neither as minimal as storing just the *TLD* (which results in many *False Positives*) nor as detailed as collecting every *URI-R* present in every archive (which goes stale very quickly and is difficult to maintain). In our experiments, we correctly identified about 78% of the *URIs* that were or were not present in the archive with less than 1% relative cost as compared to the complete knowledge profile and identified 94% *URIs* with less than 10% relative cost without any *False Negatives*. We further investigated the possibility of content-based profiling by issuing fulltext search queries (when available) and observing returned results [35] if access to the *CDX* data is not possible. We were able to make routing decisions of 80% of the requests correctly while maintaining about 90% *Recall* by discovering only 10% of the archive holdings and generating a profile that costs less than 1% of the complete knowledge profile. Furthermore, we introduced *MementoMap*, a framework for archive profile serialization and dissemination [36]. All this research primarily focuses on what is present in different web archives, while in this work we explore ways to identify what is absent from archives.

### III. SOURCES OF TRUTH

The *URI* space is infinite and the web is vast. Many people have attempted to estimate the size of the web at different times and have come up with different numbers from a few billions to a few trillions [37], [38], [39], [40], [41], [42]. However, knowing the size of the web and web archive holdings can only lead us to estimating how much of the web is not archived. If we want to know what sections of the web are not archived, we need to know all the existing *URIs*, not just their count. Knowing *URIs* of all the existing resources on the web or creating a representative sample of the web is hard. However, we can sample *URIs* from certain sources (e.g., *DMOZ*, social media, or access logs) that are of interest for a specific application, while knowing that these samples will have their own purposes and biases. We can create archive profiles of *Archival Voids* in the following ways:

- Perform lookups of sample *URIs* in an archive and record all the *URIs* that are not archived.
- Use access logs of a *Memento Aggregator* or the archive itself to identify resources that are absent from an archive.
- Use URLs from the access control lists (*ACL*), approved take down requests, resources blocked by *robots.txt* [43], and domains/*TLDs* blocked by an archive’s policy.

*URIs* collected by the means listed above can be summarized to form *Archival Voids* profile. In our prior work

```
172.17.0.1 - - [13/Nov/2020:19:01:18 +0000] "GET /favicon.ico HTTP/1.1" 200 238 "http://localhost/" "Mozilla/5.0 (X11;
Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.66 Safari/537.36"
```

Fig. 1: A Sample Extended Access Log Entry (Fields: *IP* address of the client, user identity, authenticated user’s *ID*, date and time, *HTTP* method, request path, *HTTP* version, *HTTP* status code, size of the response in bytes, referrer, and user-agent)

TABLE I: *Arquivo.pt* Access Logs Summary

Feature	Value
Number of files (1 file per day)	2,220
Total size	461G
Total size (GZipped)	37G
Total lines (requests)	1,647,573,303
Logs start date	2013-12-02
Logs end date	2019-12-31
Missing date (filled with an empty file)	2016-09-08
<i>Memento</i> support start date	2016-06-03
Log configuration changed (a field added)	2019-09-17
Major replay system upgrade (fixed many issues)	2019-11-18
<i>TimeMap</i> endpoint changed	2019-11-18

we described the *Random Searcher Model (RSM)* to learn about the holdings of an archive using fulltext search [35]. However, fulltext searching is not a suitable technique for *Archival Voids* detection because it only returns resources that are present in an archive. In this work, we only investigate *Archival Voids* profiling using access logs of an archive to learn about frequently accessed resources that are not present in the archive. Many *HTTP* servers write logs in the standard *Common Log Format* or extended *Combined Log Format* [44]. Figure 1 illustrates a typical access log file entry. In this work, we leverage access logs of web archives to learn about frequently accessed resources that are not present in the archive. To process web archive access logs we implemented a generic *HTTP* access log parser with added capabilities for web archives [45]. Approaches other than access log processing are either inefficient or beyond our abilities (e.g., we do not have access to archiving policies or *ACLs* of any web archive).

#### IV. EVALUATION

To evaluate our process for estimating *Archival Voids*, we use access logs of a web archive. We extract *URI-Rs* from the access logs and identify *URI-Rs* that have always returned “404 Not Found” responses (ignoring any “3xx” or “5xx” responses). Then we exclude *URIs* that are not accessed frequently, so that we profile only the popular resources. Figure 2 shows that there are many *URIs* that are accessed frequently, but are not archived. Being able to summarize them can significantly improve routing efficiency.

##### A. Access Logs Dataset

With the generous support from *Arquivo.pt*, we have access to over six years of their web archives’ access logs. Table I summarizes the access logs data we acquired. These log files contain about 1.6 billion records, but not all of these records are *Memento* related. *Memento* support was added to *Arquivo.pt* in June 2016.

Mementos in Arquivo.pt	10Ks+	114	0	0	0	0	0
	Ks	30.9K	37	3	5	3	0
	100s	2.4M	322	59	13	3	0
	Tens	30.2M	3.2K	372	45	1	0
	Ones	2.0B	48.4K	1.1K	88	6	1
	Zero	N/A	3.2M	19.0K	863	43	2
		Zero	Ones	Tens	100s	Ks	10Ks+
Requests for URI-Rs in MemGator Logs							

Fig. 2: Overlap between archived and accessed resources in *Arquivo.pt*. *Ones* denote single digit non-zero numbers (i.e., 1–9), *Tens* denote two digit numbers (i.e., 10–99), and so on. The *Zero* column shows the number of mementos of various *URI-Rs* that are never accessed using *MemGator*. The *Zero* row shows the number of access requests for various *URI-Rs* using *MemGator* that are not archived. The (*Zero*, *Zero*) cell denotes *N/A* because the number of resources that are neither archived nor accessed is unknown.

##### B. Access Patterns

Figure 3 illustrates daily access patterns of *Arquivo.pt*. There is a significant increase in traffic for last few months of a 2017 and for the most part of 2018. On further investigation on user-agents and status code distribution, we found that this increase in traffic was primarily from *Googlebot* and a small portion of it was coming from *YandexBot*. Together, these two bots were responsible for over 80% of the traffic.

On November 23, 2018, *Arquivo.pt* updated their *robots.txt* file to exclude all the bots from accessing their resources under the “/wayback” path under which their archival replay operates. In Figure 4 we illustrate two versions of their *robots.txt* from the same day, on which the latter shows corresponding change made to the file. The timing of this change corresponds to the drop in traffic coming from search engine bots.

Furthermore, we noticed an increased bot activity in 2019 that attempt to access *Arquivo.pt*’s *robots.txt* file many times every second. These requests are coming from many different locations, and some of those hosts belong to *Google*.

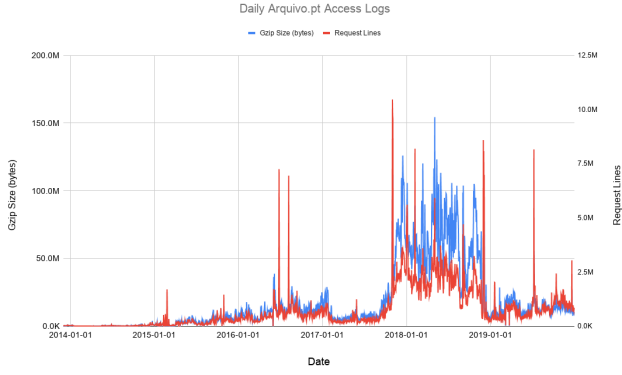


Fig. 3: Access Patterns in Six Years of *Arquivo.pt*'s Log Files

```

1 $ curl https://web.archive.org/web/20181123104043id_
2   ↪ https://arquivo.pt/robots.txt
3 User-agent: Arquivo-web-crawler
4 Disallow: /wayback
5
6 User-agent: *
7 Disallow: /nutchwax/search
8 Disallow: /search
9
10 $ curl https://web.archive.org/web/20181123125853id_
11   ↪ https://arquivo.pt/robots.txt
12 User-agent: Arquivo-web-crawler
13 Disallow: /wayback
14 Disallow: /noFrame/replay
15
16 User-agent: *
17 Disallow: /wayback
18 Disallow: /noFrame/replay
19 Disallow: /nutchwax/search
20 Disallow: /search

```

Fig. 4: *Arquivo.pt* Excluded Bots from Accessing Its Archival Replay on November 23, 2018

They all have the same request signature (i.e., the same request *URI*, user-agent, and referrer). While we do not fully know the purpose and origin of these requests yet, it does not concern us much because the requests are not about mementos or their replay system, so they are out of our scope for this work.

Table II shows most frequent *URIs* that were accessed at least 10,000 times from *Arquivo.pt* over the period of six years. While their own domain *fccn.pt* and some other globally popular websites are present in this list, the high frequency of some less obvious resources suggest that they are perhaps coming from some browser add-ons or some pages that some people/tools open often where these resources are embedded.

Table III describes how often resources from various *TLDs* were accessed from the archive each year. The top five *TLDs* include “.pt”, “.com”, “.org”, “.net”, and “.eu”. When preparing these statistics, we removed any *TLDs* that did not appear in all years as they were insignificant and often malformed entries. This table only shows statistics on requests that are related to a memento (i.e., they have a *URI-R* in their path). Such requests can be *URI-Ms* (i.e., *Memento URIs*), *URI-Gs* (i.e., *TimeGate URIs*), or *URI-Ts* (i.e., *TimeMap URIs*). The grand total is a little over one billion requests, which is two thirds of the total number of requests in their logs. Numbers under the 2018 column are larger than other years

TABLE II: Frequently Accessed Resources from *Arquivo.pt*

URI	Count
<a href="https://fccn.pt/">fccn.pt/</a>	102,953
<a href="https://google.com/">google.com/</a>	44,673
<a href="https://youtube.com/">youtube.com/</a>	29,418
<a href="https://facebook.com/">facebook.com/</a>	16,778
<a href="https://connect.facebook.net/en_us/sdk.js">connect.facebook.net/en_us/sdk.js</a>	16,462
<a href="https://discovery.dundee.ac.uk/.../contributiontojournaleditor.xhtml">discovery.dundee.ac.uk/.../contributiontojournaleditor.xhtml</a>	14,608
<a href="https://tripadvisor.com.tr/cookiepingback?early=true">tripadvisor.com.tr/cookiepingback?early=true</a>	13,556
<a href="https://publico.pt/">publico.pt/</a>	13,022
<a href="https://lamonitor.com/">lamonitor.com/</a>	11,901
<a href="https://static.tacdn.com/.../bounceusertracking-v21915390943b.js">static.tacdn.com/.../bounceusertracking-v21915390943b.js</a>	11,781
<a href="https://static.tacdn.com/.../bounceusertracking-v21915390943a.js">static.tacdn.com/.../bounceusertracking-v21915390943a.js</a>	11,041
<a href="https://youtube.com/watch">youtube.com/watch</a>	10,563

due to increased activity from search engine bots in the year 2018. The “.au” *TLD* shows an interesting trend as it was not as popular as some of the other *TLDs* below it, but search engine bots seemed more interested in it in the year 2018, which made it go significantly up in the table. This suggests the need for periodic updates of *Archival Voids* profiles as the demand of certain sections of the web changes over time.

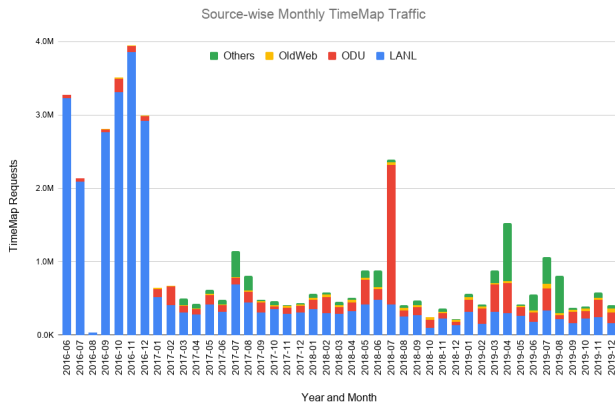
Considering *TimeMaps* being one of the most accessed resources by *Memento Aggregators*, we decided to see what other user-agents are interested in them. There were about 42 million *TimeMap* requests in their logs. We found that *LANL's TimeTravel* service is the largest source of traffic to *Arquivo.pt's TimeMap* endpoint (as shown in Figure 5). The first few months after *Arquivo.pt* added *Memento* support *LANL's Aggregator* was making a significant number of requests, but it slowed down after a few months. For the first few months *LANL* was using an old *Memento Aggregator* written in Java, which was later replaced by a new code that utilizes a classifier and an improved caching stack. The increased traffic during the first few months was likely caused by the cache front-loading and data collection for training the classifier. The second source of traffic was our own *MemGator* [46], [47] instance, running at Old Dominion University. There is a spike in July 2018, from our service, because someone used our service to access *TimeMaps* of a long list of *URIs*, which changed our regular usage pattern significantly. *OldWeb.today* uses our *MemGator* tool on its own servers to reconstruct mementos in old browsers, which was another significant source of traffic to *TimeMaps*. Other sources include a variety of user-agents, often pointing to *cURL*, *HTTP* libraries in different languages, or research projects. Two notable user-agents among them which caused increased traffic on certain months pointed to an in-house script of *Arquivo.pt* (which we believe they use for periodic service quality/health check) and a *MediaWiki* bot, called *WaybackMedic* [48], that fixes broken links.

### C. Soft-404 *TimeMaps*

As per the *HTTP* standards, if a resource is accessed that is not present on the requested *URI*, the server should return the “404 Not Found” status code, and if the resource is present and is accessible then the response should be “200 OK”. However, some poorly written web applications may

TABLE III: Yearly Access Frequency of Top TLDs in *Arquivo.pt*

TLD	2014	2015	2016	2017	2018	2019	Total
.pt	1,769,211	5,638,317	42,105,411	139,685,874	455,617,209	75,585,184	720,401,206
.com	188,937	1,041,797	12,113,651	84,960,808	122,007,623	26,912,173	247,224,989
.org	14,594	76,424	1,106,144	5,383,964	35,025,928	2,411,604	44,018,658
.net	17,035	74,222	167,770	7,421,125	18,903,068	2,938,864	29,522,084
.eu	1,089	19,539	388,764	2,727,154	20,129,910	1,346,343	24,612,799
.au	10	189	815	30,275	4,147,213	92,882	4,271,384
.gov	172	3,258	16,502	485,699	1,930,423	479,466	2,915,520
.uk	5,061	3,920	14,400	391,839	1,364,946	343,796	2,123,962
.edu	627	5,945	16,230	192,558	1,385,982	291,992	1,893,334
.br	5,901	39,667	329,656	132,944	1,030,141	266,208	1,804,517
.ru	442	637	2,666	113,716	1,179,413	95,298	1,392,172
.de	564	2,613	22,047	143,661	737,228	444,187	1,350,300
.io	9	1,501	24,598	46,006	1,150,983	79,497	1,302,594
.pl	2	743	5,787	61,107	1,071,524	116,270	1,255,433
.int	160	894	2,617	97,603	731,551	204,840	1,037,665
.fr	149	3,009	19,283	142,771	644,848	195,730	1,005,790
OTHERS	34,717	125,950	171,235	721,084	3,335,357	1,441,533	5,829,876
ALL	2,038,680	7,038,625	56,507,576	242,738,188	670,393,347	113,245,867	1,091,962,283

Fig. 5: Source-wise Monthly *TimeMap* Access of *Arquivo.pt*

```

1 $ curl -i https://example.com/absent.html
2 HTTP/1.1 200 OK
3 Content-Type: text/plain
4 Date: Thu, 08 Aug 2019 21:13:04 GMT
5 Server: Apache
6 Content-Length: 40
7
8 Sorry, the requested page was not found!

```

Fig. 6: A Sample *Soft-404* Response

return “200 OK” status code even for resources that are not present (as illustrated in Figure 6). They often advertise the unavailability of the resource via the response body instead of the status code. This behavior is called *Soft-404* [49]. Moreover, the term *Soft-404* is commonly used as an umbrella term for any error page that is returned with the “200 OK” response code due to the prevalence of “404 Not Found” errors over other error pages on the web.

To build an *Archival Voids* profile from access logs, it is important to isolate records that have never been “200 OK”. When we tried to see the distribution of *TimeMap* responses over different status codes, we found an insignificant number of “404s”, except in the last two months. This was counter intuitive because our *MemGator* logs suggest that in more than

96% requests *Arquivo.pt* returns no mementos in its *TimeMap*. After further investigation, we found that their old replay system had bugs, causing it to return *Soft-404 TimeMaps*, until it was upgraded on November 18, 2019.

Now we had two choices, either profile only the last six weeks of data or somehow identify *Soft-404* responses. Logs do not contain the response body, so we could not do much about classifying responses. However, these access logs contain number bytes they returned in each response. We could think of two possibilities of what their *TimeMap* response might have been for resources they do not have any memento of: 1) there could be a plain message saying something like “the resource is not found”, or 2) the response included the *URI-R* one or more times along with some other template body. In the first case, number of bytes will be exactly the same for all the failed *TimeMap* requests, but we did not see a single byte size over-represented. In the second case, number of bytes in response will be a linear function of the size of the request *URI* as shown in Equation 1. In this equation  $K$  represents the number of times *URI-R* appeared in the response and  $C$  is the constant size of the template body.

$$\text{Response Bytes} = K * \text{URI-R Size} + C \quad (1)$$

To investigate our hypothesis we checked for *TimeMap* requests in December 2019 access logs (when the *Soft-404* issue was fixed) to find resources that are consistently returning 404s and checked responses corresponding to them in the past logs. Table IV shows *Soft-404* records of one such resource. In this table all the rows have a consistent number of bytes (222) except the second one (225). However, the second row also has a trailing forward slash in its *URI-R*, which is missing from the other rows. This was a clear indication that the *URI-R* was repeated three times in the *Soft-404* response body, which increased the byte size of the response by 3 when only one extra character was added to the *URI-R*. Now, we knew the value of  $K = 3$  and the size of *URI-R*; using Equation 1 we can compute  $C = 150$ . With this insight, we tried it on some other requests and found it working, which gave us more confidence.



TABLE IV: *Soft-404 TimeMap* Response Bytes

Timestamp	Request	Bytes
1546885931	/wayback/timemap/*/http://matkelly.com/wail	222
1546885968	/wayback/timemap/*/http://matkelly.com/wail/	225
1547238957	/wayback/timemap/*/http://matkelly.com/wail	222
1547239466	/wayback/timemap/*/http://matkelly.com/wail	222
1547239877	/wayback/timemap/*/http://matkelly.com/wail	222

In Figure 7 we tried to reconstruct what the *Soft-404* response might have been, one that matches our calculated numbers and looks like a reasonable representation. From this response we think we know the nature of the bug in their code. They were perhaps not checking for the existence of any mementos for a given *URI-R* before generating the response. Instead they were creating the obvious initial lines of the response and then looping over all the mementos, which would loop zero times if there were no mementos and only the initial lines will be returned.

With this ability to identify *Soft-404s* reliably, we went through all the *TimeMap* requests and fixed the status code (“404” for “200”) in a copy of logs. We used these amended logs for further analysis.

#### D. Status Code Changes Over Time

To ensure that we only profile *URIs* that never returned a successful response (after amending *Soft-404s*), we decided to investigate how often *URIs* change from one status code to the other. In the case of a *URI-M* there are many status codes possible, both due to observed status codes from the origin and the state of the replay server. However, in the case of a *TimeMap* we anticipated a limited number of different status codes. Actual status code distribution of *TimeMaps* is shown in Table V. If there are no mementos for the given *URI-R*, the status should be “404”, otherwise “200”. In rare cases we expect “5xx” status codes, in case the server is facing any issues. However, the last two months of data had many “302” responses as well. On further investigation we found that when *Arquivo.pt* upgraded their replay system, they also changed some of their service endpoints (in this case, their *TimeMap* changed from “/timemap/\*/” to “/timemap/link/”), for which they put redirects in place. In addition to this, they also had a few “301” responses for certain *TimeMaps* where the *URI-R* contained *Facebook*’s tracking token in the query parameter, which they redirected to a *URI-R* without the tracking token. After knowing this, we removed all the redirect responses because they were not adding anything to our assessment of the popularity or unavailability of resources.

After this cleanup we sorted entries primarily on their canonical representation and a secondary sort on their time, so that we can know how each *URI* changed from one status to the other. We were expecting that a few resources that were “404” before would become “200” when they are eventually crawled and are made available and a few resources might go the other way if they are taken down for some reason. Other status code transformations were expected to be less likely (e.g., the server returning “5xx” response occasionally).

TABLE V: Status Code Distribution of *TimeMaps* in *Arquivo.pt* Access Logs

Status	Requests
200	2,614,615
301	2,455
302	224,535
400	98,267
404	38,615,290
429	42,720
500	134,858
503	1,015
<b>TOTAL</b>	<b>41,733,755</b>

TABLE VI: Status Code Fluctuations of *URI-Rs* in *Arquivo.pt* Access Logs

Status Codes Over Time	<i>URI-Rs</i>
404	15,502,081
200	680,328
404,200	36,447
200,404	685
200,404,200	648
404,200,404	48
404,200,404,200	43
200,404,200,404,200++	40

However, when we analyzed our data, we found that there were many fluctuations between “200” and “404”, where some resources changed their status codes back and forth hundreds of times. It turned out that it was caused by lack of proper *URI* normalization/canonicalization [50], [51]. For example, when a *TimeMap* was requested for “apple.com” they returned “200”, but for “Apple.com” or “APPLE.COM” they returned “404” instead. We thought about a few approaches to amend this effect as well, but that could change our result in ways that can be harmful, so we decided to exclude all the requests that include any upper case letter in the hostname portion of their *URI-R* and started over. After excluding entries with any upper case letter in their hostname the number of fluctuations went down, but there were still many entries with hundreds of fluctuations back and forth between “200” and “404”. We concluded that lack of *URI* canonicalization was not limited to just hostnames in *Arquivo.pt*, but perhaps they had little to no canonicalization in place.

After that we decided to work with the dataset without any canonicalization or filtering, considering each *URI-R* in the logs as an independent resource. This means we will have many non-canonical *URIs* that will always report “404” while their corresponding canonical version may or may not behave the same way. This may increase the size of our voids profiles, but we expect the prevalence of many unique non-canonical *URI-Rs* to be small, which may fall below the threshold to be included.

Table VI summarizes status code fluctuations of *URI-Rs* in the access logs of *Arquivo.pt*. There are 15,502,081 unique non-canonical *URIs* that have always returned the “404” status code and 680,328 *URIs* have always returned the “200” status code. We believe that the number of *URIs* returning “200” status code would have been a little larger and “404” status code a little smaller if *Arquivo.pt* were to exercise *URI*

```

1 <https://arquivo.pt/wayback/timemap/*/http://matkelly.com/wail>; rel="self"; type="application/link-format",
2 <https://arquivo.pt/wayback/http://matkelly.com/wail>; rel="timegate",
3 <http://matkelly.com/wail>; rel="original"

```

Fig. 7: A Potential *Soft-404 TimeMap*

canonicalization from the beginning. There were 36,447 *URIs* that returned “404” status code in the past, but later started to return “200” while there were only 685 *URIs* that gone from “200” to “404”. These numbers confirm our intuition about more *URIs* becoming available over time while a few of the existing resources disappearing (for example, blocked or taken down after reports or policy reviews). This table does not reflect how many times and for how long certain status codes remained associated with a given *URI*.

While analyzing data for status code fluctuations without any *URI* canonicalization we found that one specific *URI* was still exhibiting about 150 fluctuations back and forth between “200” and “404” status codes. On further investigation we found that it was <http://www.fccn.pt/> (this domain belongs to *Arquivo.pt*) which appeared a total of 102,799 times in the access log and 88,807 times with status codes “200” or “404”. This *URI* returned “200” status code only 105 times while “404” status code 88,702 times. We further investigated the status code fluctuation pattern for this *URI* and found that it would return “404” status code hundreds of times in a row with occasional “200” status code every once in a while. It turned out that the server always returned the “404” status code for requests coming from a specific IP address which has the “Mozilla/5.0+(compatible; UptimeRobot/2.0; http://www.uptimerobot.com/)” user-agent, but the “200” status code to everyone else (such as *MemGator* or the *TimeTravel* services). From the user-agent string we can tell it is a server health check service which periodically polls specific resources, but we do not know why the server behaves differently for this user-agent.

#### E. Routing Accuracy

After we identified most frequently accessed resources that have never returned a successful response for any of their canonical or non-canonical *URI-Rs*, we created *Archival Voids* profiles with these. Table VII shows the repetition breakdown of the number of *URIs* that have always returned the “404” status code. There are over 13 million canonicalized *URIs* that have always returned the “404” status code, but each of them appeared only 1–9 times while about 0.7 million “404” *URIs* appeared 10–99 times. The long-tail of low frequency *URIs* are not suitable for profiling voids as they will increase the size of the profile disproportionately. For example, to go from the request savings of 8.42% to 64.67% would require an increase of about four orders of magnitude in the number of *URIs* in an *Archival Voids* profile. An attempt to use less detailed profiling policies to reduce the size of the profile would introduce *False Negatives*. However, the last few rows of the Table VII represent only a few *URIs* that have been

TABLE VII: 404-Only *URI-R* Repetitions in *Arquivo.pt* Access Logs and *False Positives* Reduction Due to the *Archival Voids* Profile

Repetitions	<i>URI-Rs</i>	<i>MemGator</i> Requests Saving %
1s	13,673,599	64.67
10s	698,959	17.00
100s	2,319	8.42
1,000s	99	2.85
10,000s	2	0.00

requested thousands or tens of thousands of times and have always returned the “404” status code. Creating a voids profile with these would cut the *False Positives* down significantly.

There are over seven million entries in the *Arquivo.pt* access logs that originated from the *MemGator* server running at the *Old Dominion University*. We analyzed the percentage of requests that could have been avoided if *Archival Voids* profiles of various frequencies were made available based on the access log alone. Table VII shows that about 2.85% *False Positives* could have been avoided by only profiling *URIs* that have appeared thousands of times and have always returned the “404” status code. This saving could have been around 8.42% if we included *URIs* that were repeated more than hundreds of times. We have reported lower bounds to avoid any *False Negatives* while we believe that the numbers would have been even better if *Arquivo.pt* had a proper *URI* canonicalization in place from the beginning.

#### V. WHO SHOULD PROFILE ARCHIVAL VOIDS?

It is important to keep the profile of *Archival Voids* fresh, otherwise *False Negatives* will increase very quickly. Unlike *Archival Holdings* profiles, aggressively reducing the *URI* key size can be harmful in *Archival Voids* profiles as users will fail to discover many resources that are present in a web archive.

An *Archival Voids* profile is expected to complement an *Archival Holdings* profile, so the entries about what is missing can be very specific. However, it is possible to use an *Archival Voids* profile independently, and is ideal for large web archives such as *IA*. If an archive is going to return good results for most of the requests, then it will be wasteful to profile its holdings for the sake of routing. Knowing what it does not contain or is not willing to serve is a more compact way to improve routing *Accuracy* for such web archives.

In 2015, a Twitter bot called *ICanHazMemento* was launched which polls Twitter periodically to fetch new Tweets that contain the hashtag “#ICanHazMemento” and a *URI* in their conversation chain and replies to them with a *URI-M*, pointing to a memento of the *URI* in a web archive [52]. To find a suitable *URI-M*, it would perform a lookup in *LANL’s TimeTravel* service and link back to it. If it does not find any



mementos for the *URI-R*, it attempts to save the resource in one or more archives and then tweet about them. However, the *TimeTravel* service had caching in place, which continued to return a “404 Not Found” response (until the cache expires) despite the newly created mementos. Consequently, users following the link posted as a reply to their tweets will fail to access a memento that was created. The issue was noticed and was fixed soon after by configuring the *TimeTravel* service to not cache “404” responses. An *Archival Voids* profile generated by a third party can have a similar issue.

Because of the potential danger of *False Negatives* due to stale *Archival Voids* profiles, it is recommended that an *Archival Voids* profile is generated by an entity that is close to the source of truth (e.g., a web archive itself). When third parties (e.g., a *Memento Aggregator*) generate such profiles, they should add very specific entries and should update the profile frequently. Also, they should only add resources in such profiles when they have gained enough confidence that the resource is indeed missing and has very little chance to be available anytime soon (e.g., due to successive failure responses of frequently queried resources).

## VI. RECOMMENDATIONS

Based on our assessment, we have some recommendations for those generating such profiles (most likely, web archives themselves or *Memento Aggregators*):

- Keep *Archival Voids* profile separate as a paginated resource so that it can be updated and consumed independently and more frequently (which is also a more logical approach because the data source for the holdings profile would primarily be *CDX* indexes while voids profiles will be generated using access logs and collection policies).
- Be more specific in including *URIs* in the voids profile and include shorter *URI Keys* only when the confidence is very high or a domain or *TLD* is blocked by the collection policy (e.g., pornographic *TLD* “.xxx”).
- Update frequently from the list of take down requests from domain owners or governments.
- Include only resources that are high in demand, but missing or prohibited, because listing items that no one is requesting is not going to save unnecessary traffic while it will expose more information in public and make the profile large.

On the utilization side, proper order of evaluation will be important when there are many competing *URI Keys* for the lookup *URI* in both the *Archival Holdings* profile as well as *Archival Voids* profile with different host/path depths.

## VII. CONCLUSIONS AND FUTURE WORK

In this work we defined and discussed *Archival Voids* and established a means to represent portions of *URI* spaces that are not present in web archives. With the help of examples we explained the purpose of creating *Archival Voids* profiles and illustrated how it works in conjunction with the *Archival Holdings* profile in a hierarchical manner to describe holdings and voids in more specific portions of the *URI* spaces. We discussed various sources of truth that can be used to create

*Archival Voids* profiles. For evaluation we used access logs from *Arquivo.pt* to create an *Archival Voids* profile and analyzed it against our *MemGator* access logs. In the process we described access patterns in *Arquivo.pt* and surfaced various corner cases and issues that were present in it. We discussed prevalent *Soft-404 TimeMaps* in the access logs for many years and techniques we used to remedy that in order to make a more meaningful analysis of the dataset. We discussed the distribution of *HTTP* response status codes in the access logs and reported how these status codes changed over time for various *URIs*. We evaluated the routing *Accuracy* against various *Archival Voids* profiles created from these access logs and found that we could have avoided more than 8% of the *False Positives* on top of the 60% *Accuracy* we got from profiling *Archival Holdings* in our prior work [36], if *Arquivo.pt* were to provide an *Archival Voids* profile based on *URIs* that were requested hundreds of times and never returned a success response. Finally, we discussed who should create *Archival Voids* profile and provided some guidelines based on our understanding.

The concept of *Archival Voids* we introduced can be further investigated as a crawl quality measure in the future. For example, if a crawler job is initiated with a set of seed *URIs* and is configured to collect resources within certain scopes (e.g., all the *URIs* under “.gov” and “.mil” *TLDs*) then it is desired to know how well the configured scopes were crawled and how many resources were missed. In our evaluations we took the conservative approach to not allow any *False Negatives* (i.e., we maintain a 100% *Recall*) to establish the baseline. We believe that the *Accuracy* can be increased by allowing a small amount of *False Negatives* as it would reduce the more prevalent *False Positives*. However, this hypothesis needs to be evaluated in a future work. It is worth noting that the cost of *False Positives* affects the infrastructure while the cost of *False Negatives* affects users, and for this reason we chose not to allow any *False Negatives* in our baseline evaluations.

## ACKNOWLEDGMENTS

We thank Daniel Gomes and Fernando Melo for generously sharing complete dataset of *CDX* files and access logs from *Arquivo.pt* web archive. Himarsha Jayanetti and Kritika Garg contributed to the analysis of *Arquivo.pt* access logs. This work was supported in part by the *NSF* grant *IIS-1526700*.

## REFERENCES

- [1] H. Van de Sompel, M. L. Nelson, and R. Sanderson, “HTTP Framework for Time-Based Access to Resource States – Memento,” RFC 7089, Internet Engineering Task Force, 2013.
- [2] D. Gomes, J. Miranda, and M. Costa, “A Survey on Web Archiving Initiatives,” in *Proceedings of the 15th International Conference on Theory and Practice of Digital Libraries*, ser. TPDFL ’11, 2011, pp. 408–420.
- [3] Internet Memory Foundation, “Web Archiving in Europe: A Survey Provided by the Internet Memory Foundation,” [https://web.archive.org/web/20110523234751/internetmemory.org/images/uploads/Web\\_Archiving\\_Survey.pdf](https://web.archive.org/web/20110523234751/internetmemory.org/images/uploads/Web_Archiving_Survey.pdf), 2010.
- [4] M. Ras and S. van Bussel, “Web Archiving User Survey,” [https://www.kb.nl/sites/default/files/KB\\_UserSurvey\\_Webarchive\\_EN.pdf](https://www.kb.nl/sites/default/files/KB_UserSurvey_Webarchive_EN.pdf), 2007.
- [5] G. Truman, “Web Archiving Environmental Scan,” <https://nrs.harvard.edu/urn-3:HUL.InstRepos:25658314>, 2016.

- [6] L. Gravano, C.-C. K. Chang, H. García-Molina, and A. Paepcke, "STARTS: Stanford Proposal for Internet Meta-Searching," *SIGMOD Record*, vol. 26, no. 2, pp. 207–218, 1997.
- [7] L. Liu, "Query Routing in Large-Scale Digital Library Systems," in *Proceedings of the 15th International Conference on Data Engineering*, 1999, pp. 154–163.
- [8] J. Callan, M. Connell, and A. Du, "Automatic Discovery of Language Models for Text Databases," in *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '99, 1999, pp. 479–490.
- [9] J. Lu and J. Callan, "Content-Based Retrieval in Hybrid Peer-to-Peer Networks," in *Proceedings of the 2003 ACM CIKM International Conference on Information and Knowledge Management*, ser. CIKM '03, 2003, pp. 199–206.
- [10] —, "Federated Search of Text-Based Digital Libraries in Hierarchical Peer-to-Peer Networks," in *Proceedings of the 27th European Conference on IR Research, Advances in Information Retrieval*, ser. ECIR '05, vol. 3408, 2005, pp. 52–66.
- [11] A. Sugiura and O. Etzioni, "Query Routing for Web Search Engines: Architecture and Experiments," *Computer Networks*, vol. 33, no. 1, pp. 417–429, 2000.
- [12] T. Tran and L. Zhang, "Keyword Query Routing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 2, pp. 363–375, 2014.
- [13] W. Meng, C. Yu, and K.-L. Liu, "Building Efficient and Effective Metasearch Engines," *ACM Computing Surveys (CSUR)*, vol. 34, no. 1, pp. 48–89, 2002.
- [14] M. Klusch, P. Kapahnke, S. Schulte, F. Lécué, and A. Bernstein, "Semantic Web Service Search: A Brief Survey," *Künstliche Intelligenz*, vol. 30, no. 2, pp. 139–147, 2016.
- [15] E. Greengrass, "Information Retrieval: A Survey," <https://www.csee.umbc.edu/csee/research/cadip/readings/IR.report.120600.book.pdf>, 2000.
- [16] S. Ainsworth, A. Alsum, H. SalahEldeen, M. C. Weigle, and M. L. Nelson, "How Much of the Web Is Archived?" in *Proceedings of the 11th ACM/IEEE Joint International Conference on Digital Libraries*, ser. JCDL '11, 2011, pp. 133–136.
- [17] L. M. Alkwaï, M. L. Nelson, and M. C. Weigle, "How Well Are Arabic Websites Archived?" in *Proceedings of the 15th ACM/IEEE-CE Joint Conference on Digital Libraries*, ser. JCDL '15, 2015, pp. 223–232.
- [18] —, "Comparing the Archival Rate of Arabic, English, Danish, and Korean Language Web Pages," *ACM Transactions on Information Systems*, vol. 36, no. 1, pp. 1:1–1:34, 2017.
- [19] The GDELT Project, "The Incredibly Short Lifespan of an Online News Article," <https://blog.gdelproject.org/the-incredibly-short-lifespan-of-an-online-news-article/>, 2015.
- [20] H. SalahEldeen and M. L. Nelson, "Losing My Revolution: How Many Resources Shared on Social Media Have Been Lost?" in *Proceedings of the 16th International Conference on Theory and Practice of Digital Libraries*, ser. TPDL '12, vol. 7489, 2012, pp. 125–137.
- [21] K. Leetaru, "How Much Of The Internet Does The Wayback Machine Really Archive?" <https://www.forbes.com/sites/kalevleetaru/2015/11/16/how-much-of-the-internet-does-the-wayback-machine-really-archive/>, 2015.
- [22] H. Hallak, "Why We Need Private Web Archives: Almost Two-Thirds of Web Traffic IS NOT Publicly Archivable," <https://ws-dl.blogspot.com/2018/07/2018-07-18-why-we-need-private-web.html>, 2018.
- [23] M. Kelly, M. L. Nelson, and M. C. Weigle, "A Framework for Aggregating Private and Public Web Archives," in *Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries*, ser. JCDL '18, 2018, pp. 273–282.
- [24] M. Kelly, "Aggregating Private and Public Web Archives Using the Mementity Framework," Ph.D. dissertation, Old Dominion University, 2019.
- [25] R. Sanderson, H. Van de Sompel, and M. L. Nelson, "IIPC Memento Aggregator Experiment," <http://www.netpreserve.org/sites/default/files/resources/Sanderson.pdf>, 2012.
- [26] R. Sanderson, "Global Web Archive Integration With Memento," in *Proceedings of the 12th ACM/IEEE-CS Joint Conference on Digital Libraries*, ser. JCDL '12, 2012, pp. 379–380.
- [27] M. Graham, "The Wayback Machine's Save Page Now is New and Improved," <https://blog.archive.org/2019/10/23/the-wayback-machines-save-page-now-is-new-and-improved/>, 2019.
- [28] B. Kahle, "Users Hitting 'Save Page Now' at 100 Per Second," [https://twitter.com/brewster\\_kahle/status/994380510011928578](https://twitter.com/brewster_kahle/status/994380510011928578), 2018.
- [29] A. AlSum, M. C. Weigle, M. L. Nelson, and H. Van de Sompel, "Profiling Web Archive Coverage for Top-Level Domain and Content Language," in *Proceedings of the 17th International Conference on Theory and Practice of Digital Libraries*, ser. TPDL '13, 2013, pp. 60–71.
- [30] —, "Profiling Web Archive Coverage for Top-Level Domain and Content Language," *International Journal on Digital Libraries*, vol. 14, no. 3–4, pp. 149–166, 2014.
- [31] N. Bornand, L. Balakireva, and H. Van de Sompel, "Routing Memento Requests Using Binary Classifiers," in *Proceedings of the 16th ACM/IEEE-CS Joint Conference on Digital Libraries*, ser. JCDL '16, 2016, pp. 63–72.
- [32] M. Klein, L. Balakireva, and H. Shankar, "Evaluating Memento Service Optimizations," in *Proceedings of the 19th ACM/IEEE-CS Joint Conference on Digital Libraries*, ser. JCDL '19, 2019, pp. 182–185.
- [33] S. Alam, M. L. Nelson, H. Van de Sompel, L. L. Balakireva, H. Shankar, and D. S. H. Rosenthal, "Web Archive Profiling Through CDX Summarization," in *Proceedings of the 19th International Conference on Theory and Practice of Digital Libraries*, ser. TPDL '15, 2015, pp. 3–14.
- [34] —, "Web Archive Profiling Through CDX Summarization," *International Journal on Digital Libraries*, vol. 17, no. 3, pp. 223–238, 2016.
- [35] S. Alam, M. L. Nelson, H. Van de Sompel, and D. S. H. Rosenthal, "Web Archive Profiling Through Fulltext Search," in *Proceedings of the 20th International Conference on Theory and Practice of Digital Libraries*, ser. TPDL '16, 2016, pp. 121–132.
- [36] S. Alam, M. C. Weigle, M. L. Nelson, F. Melo, D. Bicho, and D. Gomes, "MementoMap Framework for Flexible and Adaptive Web Archive Profiling," in *Proceedings of the 19th ACM/IEEE-CS Joint Conference on Digital Libraries*, ser. JCDL '19, 2019, pp. 172–181.
- [37] M. de Kunder, "WorldWideWebSize.com – The Size of the World Wide Web (The Internet)," <http://worldwidewebsize.com/>, 2018.
- [38] A. van den Bosch, T. Bogers, and M. de Kunder, "Estimating Search Engine Index Size Variability: A 9-Year Longitudinal Study," *Scientometrics*, vol. 107, no. 2, pp. 839–856, 2016.
- [39] S. Lawrence and C. L. Giles, "Searching the World Wide Web," *Science*, vol. 280, no. 5360, pp. 98–100, 1998.
- [40] K. Bharat and A. Z. Broder, "A Technique for Measuring the Relative Size and Overlap of Public Web Search Engines," *Computer Networks*, vol. 30, no. 1–7, pp. 379–388, 1998.
- [41] A. Alarifi, M. Alghamdi, M. Zarour, B. Aloqail, H. Alraqibah, K. Al-sadhan, and L. Alkwaï, "Estimating the Size of Arabic Indexed Web Content," *Scientific Research and Essays*, vol. 7, no. 28, pp. 2472–2483, 2012.
- [42] M. Khabisa and C. L. Giles, "The Number of Scholarly Documents on the Public Web," *PLOS ONE*, vol. 9, no. 5, 2014.
- [43] M. Koster, "A Method for Web Robots Control," <http://www.robotstxt.org/norobots-rfc.txt>, 1996.
- [44] Apache HTTP Server, "Common Log Format and Combined Log Format," <https://httpd.apache.org/docs/trunk/logs.html>, 2013.
- [45] S. Alam, "AccessLog Parser and CLI," <https://github.com/oduwsdl/accesslog-parser>, 2019.
- [46] S. Alam and M. L. Nelson, "MemGator - A Portable Concurrent Memento Aggregator: Cross-Platform CLI and Server Binaries in Go," in *Proceedings of the 16th ACM/IEEE-CS Joint Conference on Digital Libraries*, ser. JCDL '16, 2016, pp. 243–244.
- [47] S. Alam, "MemGator: A Memento Aggregator CLI and Server in Go," <https://github.com/oduwsdl/MemGator>, 2015.
- [48] G. Cardamom, "WaybackMedic Bot," <https://github.com/greencardamom/WaybackMedic>, 2016.
- [49] L. Meneses, R. Furuta, and F. Shipman, "Identifying 'Soft 404' Error Pages: Analyzing the Lexical Signatures of Documents in Distributed Collections," in *Proceedings of the 2nd International Conference on Theory and Practice of Digital Libraries*, ser. TPDL '12, vol. 7489, 2012, pp. 197–208.
- [50] K. Sigurðsson, "URI Canonicalization in Web Archiving," <https://kris-sigur.blogspot.com/2015/03/uri-canonicalization-in-web-archiving.html>, 2015.
- [51] M. Kelly, L. M. Alkwaï, S. Alam, M. L. Nelson, M. C. Weigle, and H. Van de Sompel, "Impact of URI Canonicalization on Memento Count," in *Proceedings of the 17th ACM/IEEE-CS Joint Conference on Digital Libraries*, ser. JCDL '17, 2017, pp. 303–304.
- [52] A. C. Nwala, "I Can Haz Memento," <https://ws-dl.blogspot.com/2015/07/2015-07-22-i-can-haz-memento.html>, 2015.