1 Introduction

Web archiving initiatives generate vast amounts of data. The Internet Archive advertise their collection to be almost 2 petabytes, growing at a rate of 20 terabytes per month[1]. As of October 2013 the British Library's archive of the UK web totalled 21 terabytes, growing by 4.5 terabytes over a one month period[2]. The cost of providing storage for large collections can be high. For instance Amazon's Glacier service, an "extremely low-cost storage service", would charge The Internet Archive $24,983 per month[3]to store their collection. Requests to browse the archive would incur additional costs. This situation motivates us to ask how web archive data can be compressed in order to optimally reduce storage space.

The Web ARChive (WARC) file format is the ISO standard[4]commonly used to store web archive data. It is a plain text format that contains records of requests and responses of URLs, along with associated metadata, such as a list of links contained within the response data. The recommendation in the WARC standard is to append records to WARC files until they reach a size limit, at which point they should be gzipped and stored. The recommendation is that uncompressed WARC files should be no larger than one gigabyte. Using this recommendation our data set from Section 4 compresses down to 28.49972% of its original size. The WARC file format is extensible and the standard lists possible compression extensions. To our knowledge no such extension has been made publicly available and none are widely used. In this paper we explore possible extensions to the WARC format that would allow delta compression of consecutive records as well as different compression algorithms. We aim to: (i) reduce the total archive size and, (ii) allow easy partitioning of the database. The strategy that leads to the smallest total archive size compresses down to 19.28690% of the original.

2 Background

1. WARC spec

[4]https://archive.org/about/faqs.php#9

[4]https://web.archive.org/web/20131017144821/http://www.webarchive.org.uk/ukwa/statistics

[4]http://calculator.s3.amazonaws.com/calc5.html#r=DUB&s=GLACIER&key=calc-8B239980-5FC4-4CFF-B8B0-21CA0A49AEE3

[4]http://www.iso.org/iso/catalogue_detail.htm?csnumber=44717

2. gzip, bzip2, zip, tar. What they do, why these ones?

3. vcdiff, bsdiff, diffe. What they do, why these ones?

4. internet archive, IIPC, BL. Collections, experience

5. Heritrix

3 Experiment: Generated Data

1. generate 1MB text data

2. apply change repeatedly

3. compression strategy

4. compare

5. very many changes, over time. What wins

6. More than just text data? What kinds of changes?

4 Experiment: GitHub Pages

The code hosting service GitHub[5]offers a free service called GitHub Pages[6]that allows users to host static web content for free. In order to evaluate different compression strategies on realistic data we conducted a crawl that collected every GitHub repository with "github.io" in the project title. GitHub projects with this naming scheme are treated as GitHub Pages projects. Their contents are compiled by the static website generator software Jekyll[7]and hosted. There were 27,507 suitable projects as of November 2013. Of these projects, X did not contain suitable web documents.

Each suitable project was downloaded as a git repository. This enabled us to iterate through every change made to the website by considering each commit, one at a time. For each commit we ran a Heritrix crawl job over the available files.

1. duplicate detection requires the hash of all previous records

[7] `http://github.com/`
[7] `http://pages.github.com/`
[7] `http://jekyllrb.com/`

2. what if you just diffed from a previous record. do you need more storage space?

3. if not you can pass an archive job around by providing a single record to diff against

4.1 Content Analysis

Run analysis over the contents of the GitHub crawl.

5 Experiment: National Archive Collection

1. Get data from major collection

2. BL, archive.org, etc.

3. Apply best strategy from previous section

4. What real-world savings can we demonstrate?

6 Conclusion

The defaults in the WARC standard do not take advantage of the fact that many documents on the web will have many minor changes made to them over time. By using a delta algorithm as well as a compression algorithm we can reduce the total archive size by nearly half again.