Introduction to Digital Libraries CS-751: Assignment #2

Due on Thursday, March 5, 2015

Michael L. Nelson 4:20pm

Avinash Gosavi

Contents

Question 1

Choose 100 URIs from A1
Generate WARC files of those URIs using:
wget
WARCreate
Heritrix (stand-alone or via WAIL)
webrecorder.io
Describe the resulting WARC files: quantitatively compare & contrast the results of the WARC files of the same URI as generated by different tools choose interesting examples
Demonstrate playback of 2-3 WARCs in the (Wayback Machine (via WAIL or stand-alone) or pywb) and (webrecorder.io)

Answer

WARC can be created using following four tools wget, WARCreate, Heritrix and webrecorder.io.

wget was the simplest to implement. It was just one line of code.

WARCreate is a chrome plugin to create a warc of current file. It was not creating WARCs for certain urls and from what I observed it was not doing it for pages that have deep links like imdb.

Heritrix warc was done using WAIL. Setting up wail took a lot of time. Wail configuration was also not straightforward. It was taking time to create WARC for certain urls as they had many sub pages. So later on a configuration was added to heritrix to restrict by time size and documents in heritrix job config file.

Webrecorder.io is an online tool for creating and replaying WARC file. It was simple to use. In Webrecorder.io was pretty useful in replaying WARC files created from all four methods.

Figures 1 to 15 are screenshot for replay of warc generated by using Heritrix, WARCreate, webrecorder.io and wget in the same order. Also after every example I am adding description and comparison for warc files. Urls for the three url are as below 1) https://twitter.com/rachidblog1/status/564533520916561923 2) http://willbrooks.deviantart.com/art/Doctor-Who-Titles-GIF-491765997 3) http://www.f1technical.net/features/19903

Figure 16, 17 & 18 are graphs comparing sizes for warc files created by different tools.

The sizes of warc files do depend on the method that we use to generate the warc file. Wget has the least size for all four URI. Heritrix would always have bigger size because it always crawls into sub-pages for a URI. The size of WARC also depends on url we are trying to call. If it has more media and deep pages it will be of bigger size.

Question 2

Ingest the 100 URIs from their resulting WARC files into a SOLR instance Demonstrate several functioning queries on the files (a full front-end is not required) describe the configuration choices you made in setting up SOLR and processing the documents

Answer

Solr has configurations to specify things like what we can index from warc files and what shouldnt be indexed. We can specify if images are to be extracted or not.

For processing documents in solr I executed following command.

```
java -jar warc-indexer-2.0.1-20150116.110435-2-jar-with-dependencies.jar \
  -s http://localhost:8080/discovery -t \
  /Users/avinashgosavi/projects/cs851-s15/assignment2/wget-combined-warc
```

For running queries I kept the default configuration given in webarchive-discovery.

Figure 19 shows a query to return urls for domain twitter. This would give links from twitter.
Figure 20 shows a query to return urls that org as public suffix. This would give me all org domain urls.

Figures

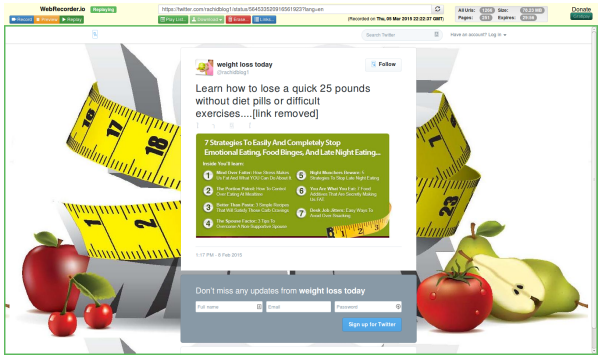

Figure 1: Heritrix

Figure 2: WARCreate

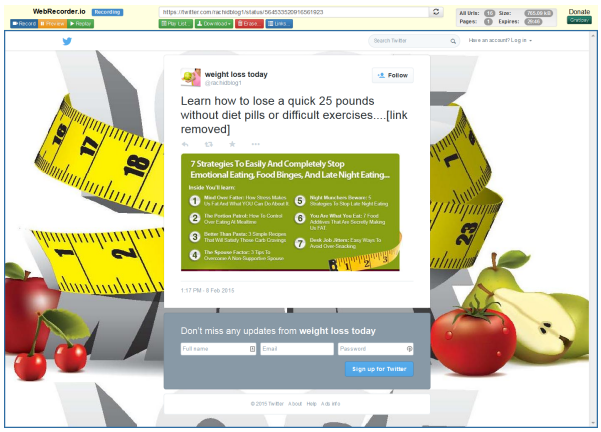

Figure 3: Webrecorder.io

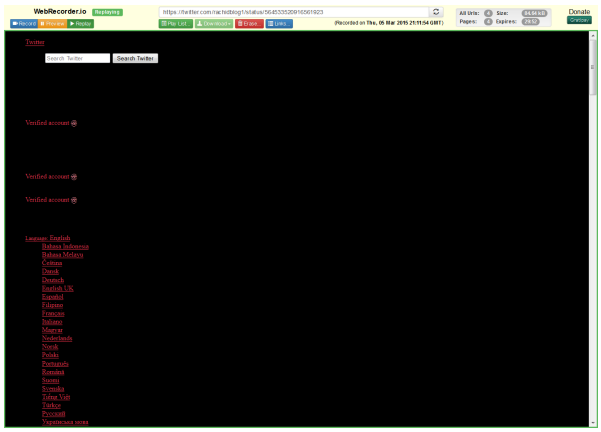

Figure 4: wget

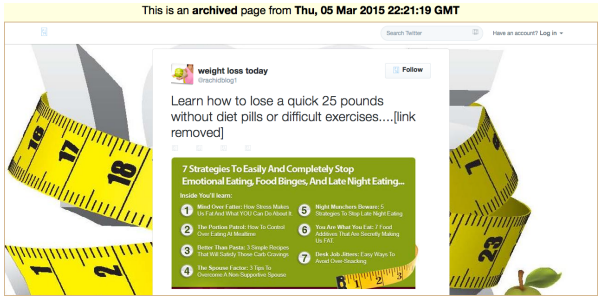

Figure 5: Heritrix - pywb

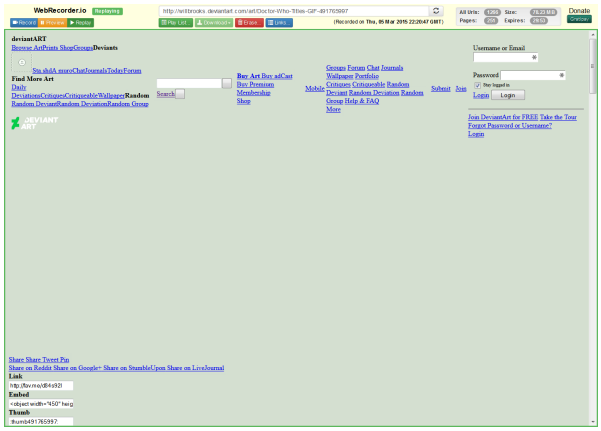

Figure 6: Heritrix

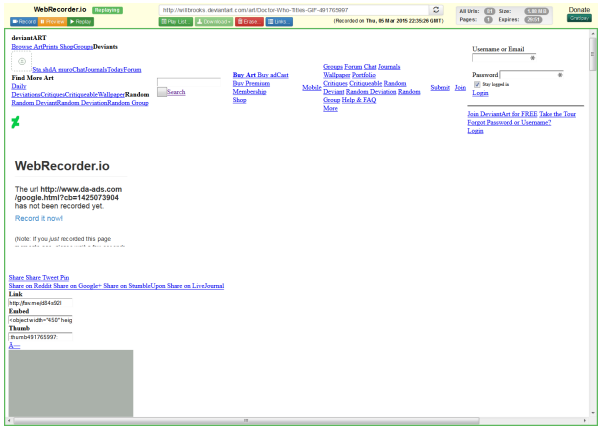

Figure 7: WARCreate

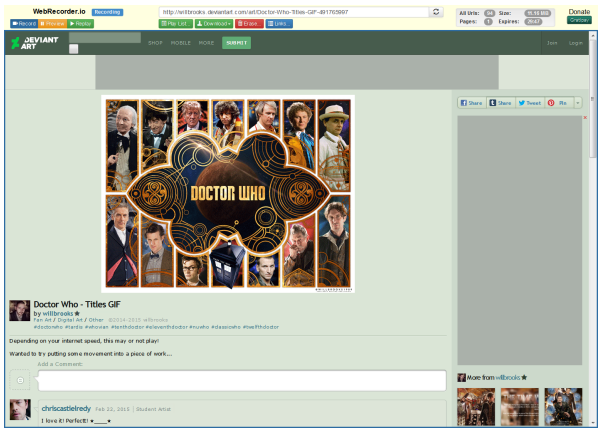

Figure 8: Webrecorder.io

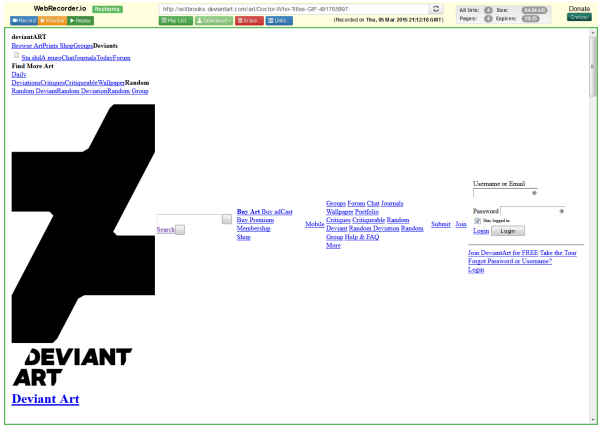

Figure 9: wget

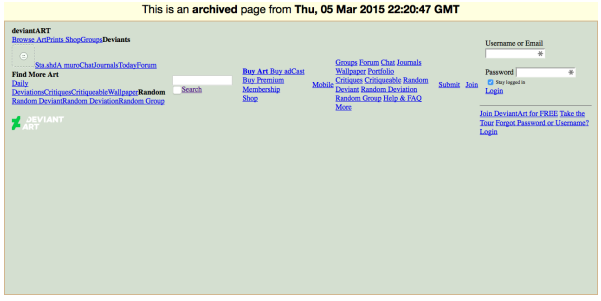

Figure 10: Heritrix - pywb

Figure 11: Heritrix

Figure 12: WARCreate

Figure 13: Webrecorder.io

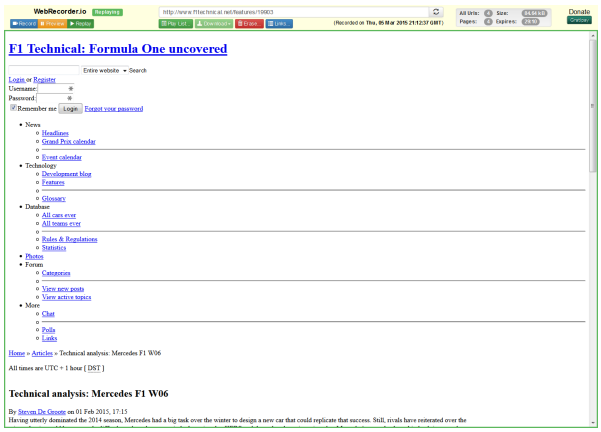

Figure 14: wget

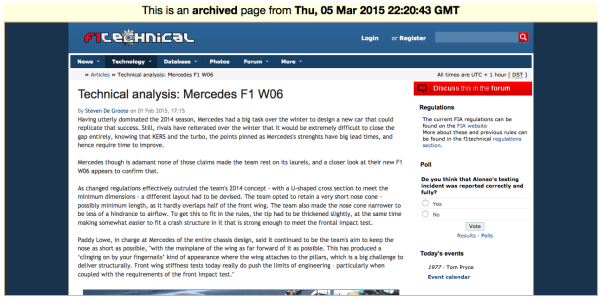

Figure 15: Heritrix - pywb

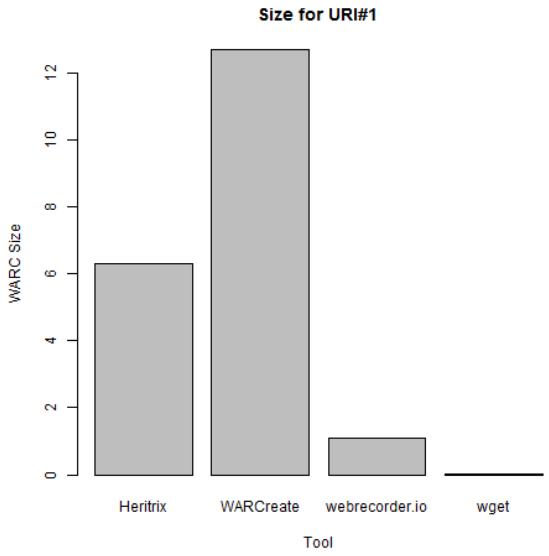

Figure 16: Graph 1

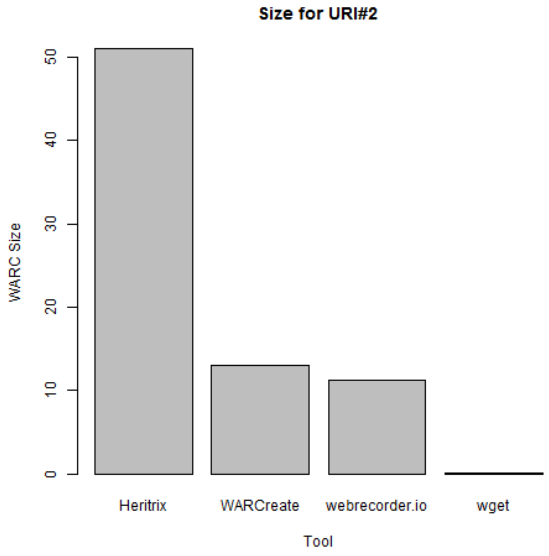

Figure 17: Graph 2

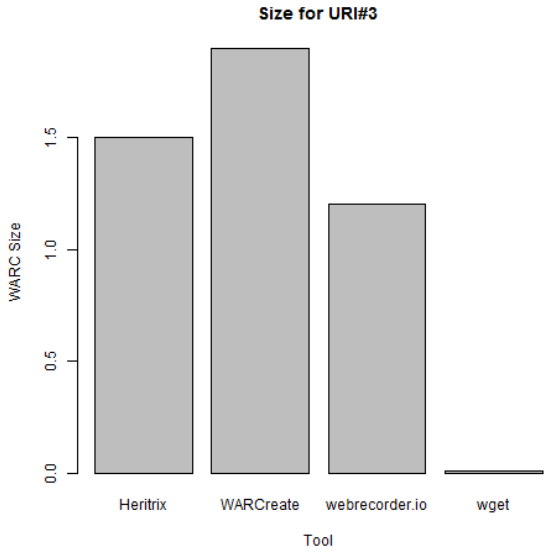

Figure 18: Graph 3

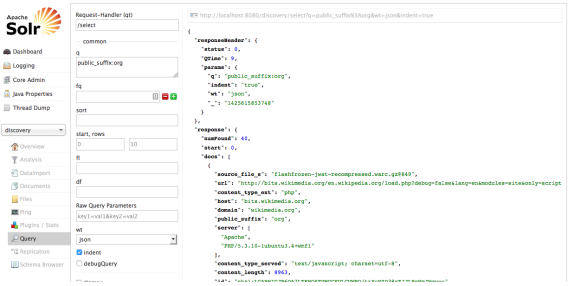

Figure 19: Solr 1

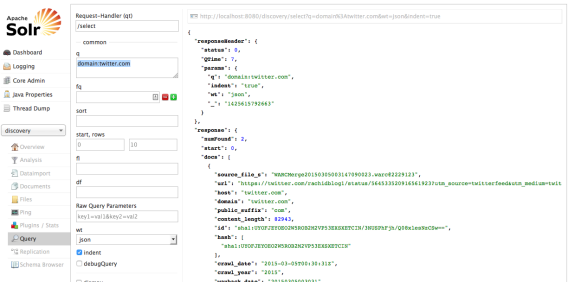

Figure 20: Solr 2