

# **Introduction to Digital Libraries: Assignment #2**

*Dr. Nelson*

**Alexander Nwala**

Thursday, March 5, 2015

## Contents

<b>Problem 1</b>	<b>3</b>
<b>Problem 2</b>	<b>8</b>

## Problem 1

Choose 100 URIs from A1 Generate WARC files of those URIs using:

1. wget
2. WARCcreate
3. Heritrix (stand-alone or via WAIL)
4. 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) <https://github.com/iipc/openwayback>, <https://github.com/ikreymer/pywb>

## SOLUTION

The solution for this problem is outlined by the following steps:

1. **Generating WARC files with wget:** wget [4] provides a functionality to create single WARC files from multiple URIs as outlined below.

```
$ wget --warc-file=outputFileName -p -l 1 URI0 URI1 ... URIN
```

Using this functionality, Listing 1. outlines the procedure of generating 2 WARC files - one file for the first 50 URIs, and the second - for the last 50 URIs. Only one level of crawl (-l 1 option) was used to generate the WARC files. The outcome of using wget was 2 WARC files of size 3.7MB and 6.7MB.

Listing 1: Generate WARC file

```
#generate warc files for multiple URIs
def downloadWarc(inputFile):
    inputFile = inputFile.strip()
5     if(len(inputFile) == 0):
        return

    lines = []
    try:
10         inputFile = open(inputFile)
        lines = inputFile.readlines()
        print '...readlines:', len(lines)
        inputFile.close()
    except:
15         print 'FILE ERROR'

    longURL = ''
    for url in lines:
        longURL = longURL + ' ' + url.strip()
20
```

```

    try:
        co = 'wget --warc-file=grandWarc2 -p -l 1' + longURL
        commands.getoutput(co)
    except:
25         print 'WGET ERROR'

```

2. **Generating WARC files with WARCreate:** WARCreate [2] provides a means to preserve a single page at a time. Consequently, in order to generate WARC files, I visited each URI from the list of 100 and clicked WARCreate's "Generate WARC" button in order to download the WARC files for the respective URIs. 60 WARC files were created from the set of 100. The outcome of using WARCreate was 60 WARC files of total size 998.8MB.

Given that WARCreate does not consolidate multiple WARC files from multiple URIs, the following command was used in order to generate a single WARC file from the set:

```
cat *.warc > *.outputFileName.warc
```

3. **Generating WARC files with Heritrix via WAIL:** WAIL (Windows version) [1] was used to crawl the set of URIs to generate a single WARC file. The outcome of using WAIL was 1 WARC file of size 120.8MB. The following outlines the procedure for configuring WAIL to generate the WARC file:

1. From windows command prompt, cmd.exe  
cd C:\WAIL
2. From C:\WAIL  
python WAIL.py
3. From WAIL GUI  
Advanced > setup-one crawl > click top empty textbox >  
add URI > write - heritrix config
4. From C:\WAIL\jobs\  
From crawler-bean.xml  
From section "URLS HERE"  
Add URIs to crawl  
From section "CRAWLLIMITENFORCER"  
Set the following properties:  
<property name="maxBytesDownload" value="100000000" />  
<property name="maxDocumentsDownload" value="10000" />  
<property name="maxTimeSeconds" value="10800" />
5. save crawler-bean.xml
6. From WAIL GUI  
From Advanced Tab  
Click Launch crawl  
Click View in Heritrix browser
7. From browser  
Click job  
Click launch

4. **Generating WARC files with webrecorder.io:** webrecorder.io [3] provides a means of archiving URIs and downloading the archived URIs into a single WARC file. Using the webrecorder.io service

Table 1: Comparison Of The Different Methods

Method	Count of WARCs	Concurrency	Ease of Use	Recommended User	Crawl scope
wget	Multiple	Yes	Medium	Advanced Archivist	Large
WARCreate	Single	No	Easy	Beginner Archivist	Small
WAIL	Multiple	Yes	Hard	Advanced Archivist	Large
webrecorder.io	Multiple	No	Easy	Beginner Archivist	Small

Table 2: Comparison Of The WARC Files

Method	Size (MB)	URIs Count	Page Count
wget	2.57	103	14
WARCreate	8.99	74	1
WAIL	11.02	158	31
webrecorder.io	13.57	68	3

online, I recorded the first URI, then continued to record others by adding new URIs and clicking on the reload button. Finally I downloaded the resultant WARC file. The outcome of using webrecorder.io was 1 WARC file of size 119.8MB

Consider Table 1. for comparison of the WARC files as generated by the different methods.

Consider the following comparison based on crawl control:

1. **wget:** wget provides a means to set the crawl level.
2. **WARCreate:** WARCreate does not expose any means to set the crawl level.
3. **WAIL:** WAIL exposes the Heritrix setting to control the crawl by setting limits on the maximum size of crawl (maxBytesDownload), the maximum number of files to download (maxDocumentsDownload), the maximum time to be spent crawling (maxTimeSeconds).
4. **webrecorder.io:** webrecorder.io enables the user to interactively explore pages to be archived.

WebRecorder.io

Replaying

Record

Preview

Replay

https://www.facebook.com/salmasoakd

Play List...

Download

Erase...

Links...

All Uris: 39

Size: 1.25 MB

Pages: 1

Expires: 29:47

Donate

Gratipay

facebook


Email or Phone

Password

☐ Keep me logged in

Forgot your password?

Log In



Lucas Salmaso

Lucas Salmaso is on Facebook.

To connect with Lucas, sign up for Facebook today.

Sign Up

Log In

Favorites

Music

Descontrol

Porta

Sabroso

LA Champions

Books

Perfeitaismo

Movies

2012

Sobreviví al 2000 al

Yo también dije la


The Extendables

Wrong Lucas Salmaso? Try Again


Lucas Salmaso

Search

Others Named Lucas Salmaso




Lucas Salmaso

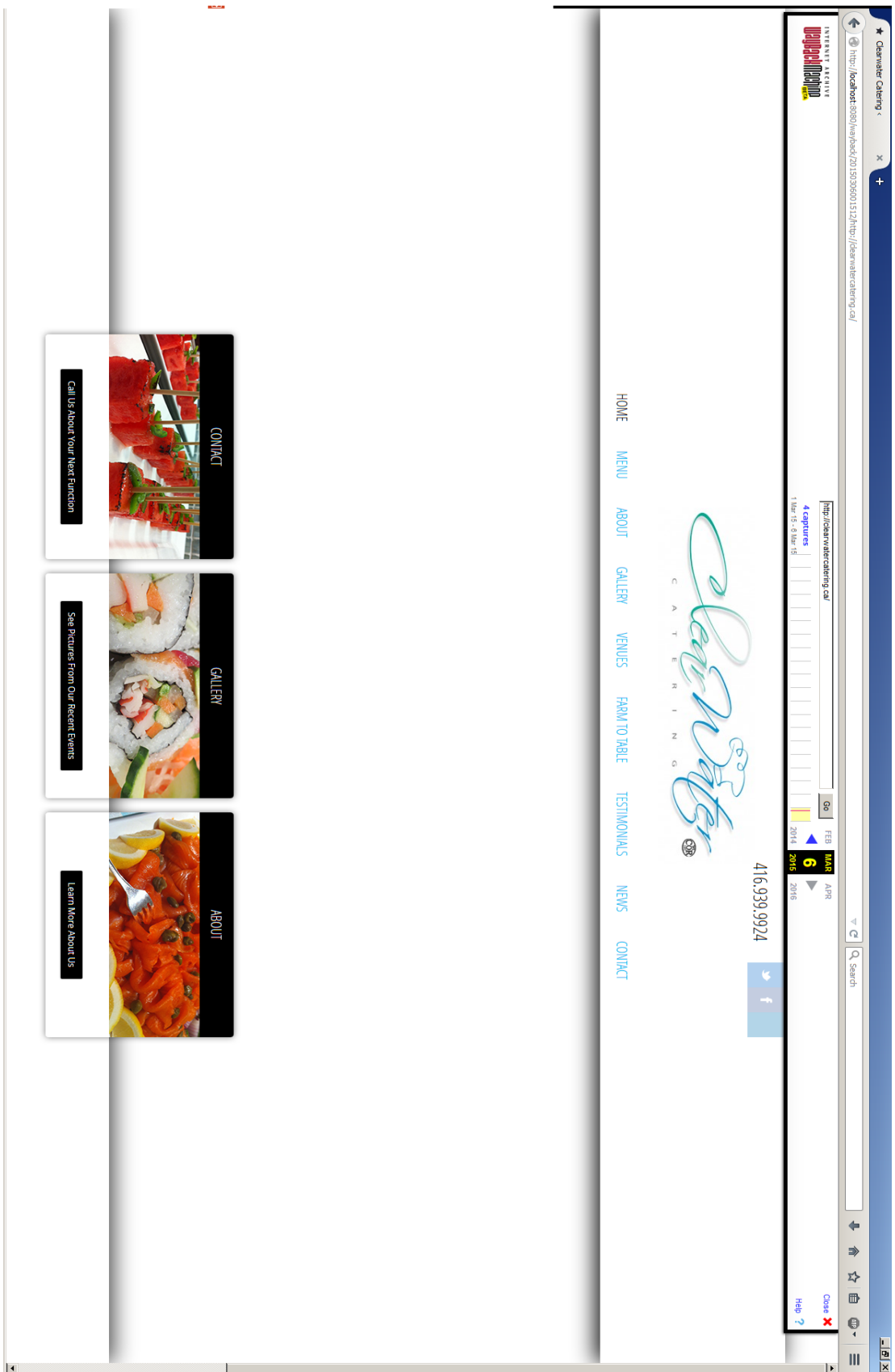


Lucas Salmaso

Others With a Similar Name



webrecoder.io playback of <http://www.facebook.com/salmasoakd/>



Wayback machine playback of <http://clearwatercatering.ca/>

## Problem 2

Ingest the 100 URIs from their resulting WARC files into a SOLR instance see the code + tutorial at: <https://github.com/ukwa/webarchive-discovery>

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

### SOLUTION

The solution for this problem is outlined by the following steps:

**SOLR Configurations:** Consider the following setting:

From solrconfig.xml: Keep the default cache size of 512

From schema.xml: Change the restrictive "AND" default solrQueryParser to "OR," in order to increase recall at the expense of Precision.

From stopwords.txt: Populate with list from <http://www.ranks.nl/stopwords>, in order to filter out stopwords

**Query 1:** Consider the excerpts of output for query - "food"

```
"source_file_s": "MAT-2.warc@77683046",
"url": "https://foursquare.com/",
...
"content": [
  "Norfolk | Food, Nightlife, EntertainmentFoursquare
  Log InSign UpHere are some popular tips in Norfolk.
  Select a taste to see more:Beer gardensHealthy foodCocktailsGood
  for workingMexican foodTrendy placesPhilly
  cheesesteaksBagelsSeafoodFoursquare
],
```

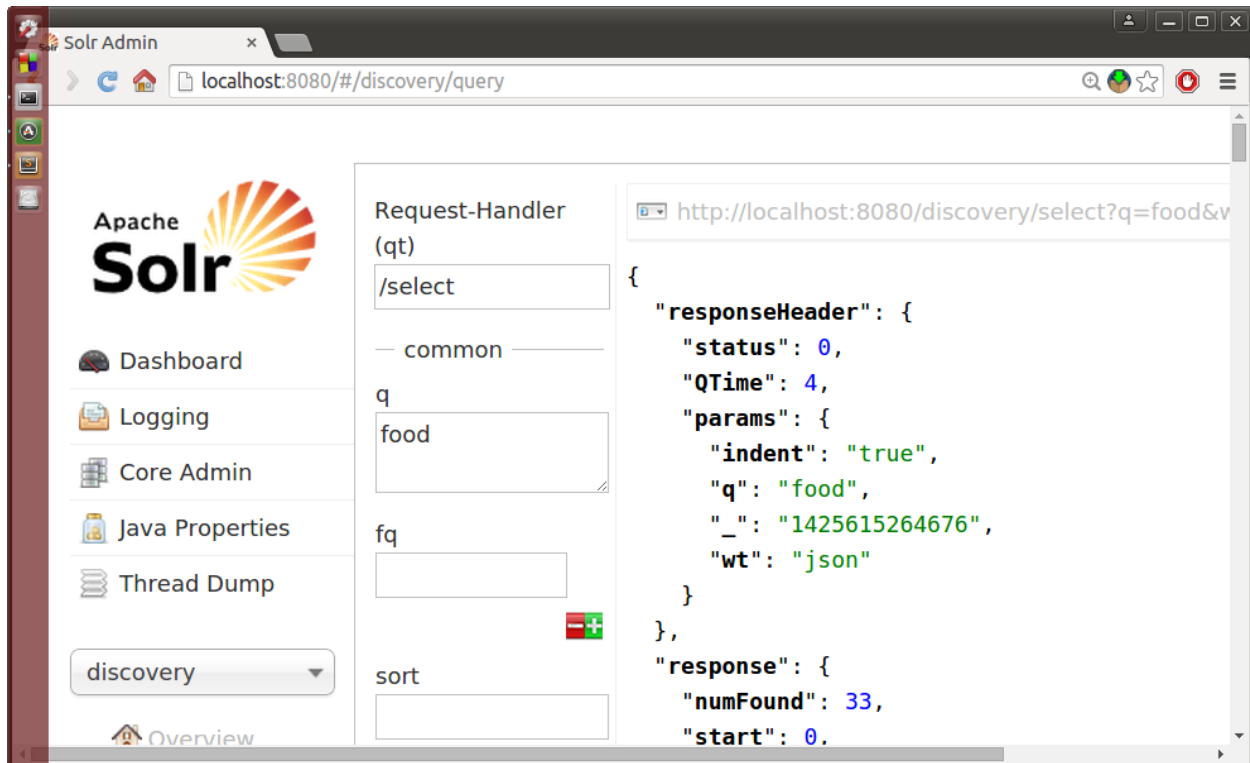
**Query 2:** Consider the excerpts of output for query - "healthcare"

```
"content": [
  "You Can't Understand ISIS If You Don't Know the History
  of Wahhabism in Saudi Arabia | Alastair... -
  Linkis.comClose panel
  http://www.huffingtonpost.com/alastair-crooke/isis-
  wahhabism-saudi-arabia_b_5717157.html?utm_hp_ref=tw4 views tweets
  You Can't Understand #ISIS
```

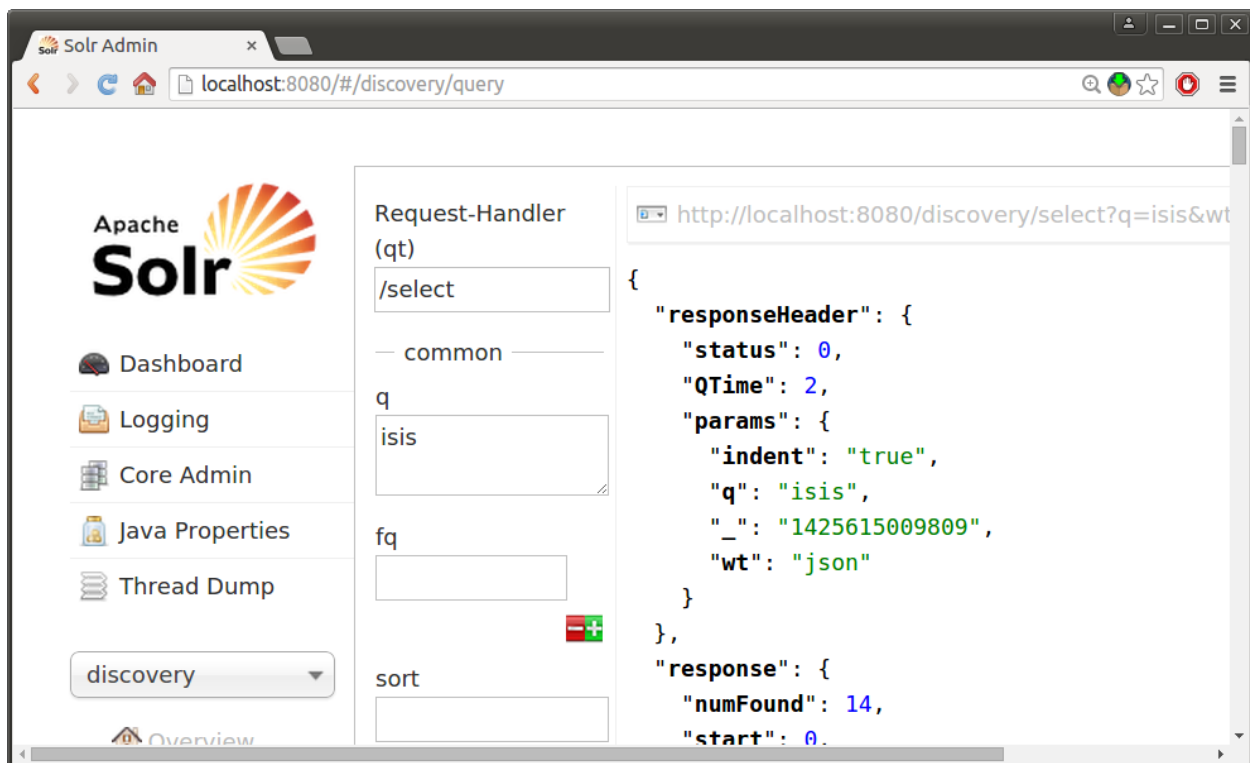
The files q1.json and q2.json contain the complete data for "food" and "healthcare" queries respectively



Figure 1: Snapshot of SOLR local server



(a) Output for Query Food



(b) Output for Query Healthcare

## References

- [1] WAIL. <http://matkelly.com/wail/>. Accessed: 2015-04-03.
- [2] WARCreate. <http://warcreate.com/>. Accessed: 2015-04-03.
- [3] webrecorder.io. <https://webrecorder.io/>. Accessed: 2015-04-04.
- [4] Wget. <https://www.gnu.org/software/wget/>. Accessed: 2015-04-03.