1. Overview

This assignment will give you an understanding of crawling, deduplicating and acquiring an extremely interesting dataset – three polar and arctic data repositories. We will focus on three polar arctic data repositories: the National Science Foundation Advanced Cooperative Arctic Data and Information System (ACADIS, upper left), the National Aeronautics and Space Administration Antarctic Master Directory (AMD, upper right) and the National Snow and Ice Data Center (NSIDC) Arctic Data Explorer (ADE, bottom). These data sets represent a rich combination of web documents, PDF documents, scientific data files (HDF, NetCDF, Gridded Binary, etc.) collected over space (world, regional, Arctic, Antarctic) and time (years-decades), with rich metadata features, and content. Access and search to this data to date has been limited to (as a baseline), scientific search of the metadata via forms; text search on the descriptions of the data; and in some (advanced) cases, geospatial search by space and time. The data also represents 10s of thousands to 100s of thousands of records, 10s-100s of Gigabytes of information, and across the three systems, which themselves are largely uncoordinated, there represents a very large chance of duplication of all types of content present in the datasets. A rough estimate is that each repository contains on order of 20,000 – 200,000 URLs; is between 10-100Gb on disk; and contains 1000-10,000 scientific records and datasets. As an idea of the richness of the types of data found in these repositories, see the

![Figure 1: NASA AMD, NSF ACADIS and NSIDC Arctic Data Explorer.](image-url)
pie chart below that compares the resultant MIME types from searching for mass balance in NASA’s Antarctic Master Directory:

![Chart](image)


2. Objective

You are going to crawl and acquire as much data from ACADIS, ADE and AMD. As much data will be defined through both web completeness (in my funded NSF project, we have actively been crawling and acquiring this data); as well as via richness of the parsed and retrieved records. To perform this, you will use Apache Nutch (http://nutch.apache.org/) that in turn will expose you to the Apache Tika (http://tika.apache.org/) framework that Nutch uses for content detection and analysis.

The ultimate objective is to find and identify and acquire as much of the content present on these repositories as possible, while also adhering to all of the things we have discussed about to date in class related to crawling, e.g., politeness, parallel crawling, all the while identifying the expanse of the web graph and the portion of the domain and graph that you are crawling. You are also required to ensure that you are not simply crawling and acquiring meaningless content on these sites – rest assured the goal is not for you to download all the javascript and CSS files as part of these repositories.

While crawling and performing the above objective, you will run into a number of issues ranging from having to negotiate different protocols to get to the actual rich content (the HDF files, the NetCDF files, the Grib files, the Matlab files, etc.) – some are behind FTP, HTTP, HTTPS, etc. You will run into issues such as some of the navigation requiring web forms and having to POST to those forms to navigate to the actual content. Some issues will relate to having to execute Javascript even, and even interpret the page in order to obtain content from it – a technique that GoogleBot and other crawlers are actively adopting as we discussed in class. So think carefully about your work and plan
before you jump head first into crawling. How are you going to deal with *politeness*? How are you going to deal with *completeness*? How are you going to deal with *dark data* and the web that exists behind *forms*? And finally how are you going to deal with *dynamic content*?

You will use Nutch as the core framework to perform crawling, and Tika as the main content detection and extraction framework. More information (installing Nutch, and Tika, etc.) will be provided in Section 4. In addition, you are going to use some under development plugins and code for Nutch and Tika that will help you understand what is required to properly deal with extraction of information from the content. You will need to extract and understand both the text, metadata, and language from the content in order to develop an algorithm to perform deduplication. You are responsible for constructing two algorithm types for deduplication – one that identifies *exact* matches, and another for *near duplicates*. This will be discussed in more detail below.

The assignment specific tasks will be specified in the following section.

### 3. Tasks

1. Download and configure Nutch to crawl AMD, ADE and ACADIS
   a. Identify and make changes to Nutch configuration to deal with politeness.
   b. Identify and make changes to Nutch configuration to deal with URL filtering.
   c. Create team identification information and label your Nutch bot via Nutch configuration.

2. Perform crawls of AMD, ADE and ACADIS
   a. Develop a program or script to iterate through your Nutch crawl data and classify what MIME types you encounter.
   b. Identify at least 15 different MIME types that you encounter while crawling.
   c. Deliver a list of at least 100 URLs that you have difficulty fetching and identify why (e.g., protocol issue, behind a web form, requires Ajax, etc.).
   d. Deliver the crawl statistics from your crawls of each repository.

3. Build the latest version of Tika and Install Tesseract, and GDAL
   c. Download and install Nutch trunk.

4. Build and install the Nutch Selenium plugin
   a. [https://github.com/momer/nutch-selenium/](https://github.com/momer/nutch-selenium/)
   b. Incorporate this into your Nutch trunk build from 3c.

5. Re-run your AMD, ADE and ACADIS crawls with the enhanced Tika and Nutch Selenium you’ve built in step 3
a. Identify at least 100 URLs that you have difficulty fetching and identify why
b. Are the URLs present from 2d still?
c. Did the enhanced Tika parsing assist with that?
d. Deliver your updated crawl statistics from each repository your crawl.

6. Develop two deduplication algorithms to use the extracted text and metadata in the Parsed Content from Nutch
   a. One for exact duplicates using the extracted Text from your AMD, ACADIS and ADE crawls.
   b. One for near duplicates using the extracted metadata from your AMD, ACADIS and ADE crawls.

7. Construct two URLFilter plugins for Nutch
   a. One for exact duplicates from 6a.
   b. One for near duplicates from 6b.
   c. Identify the number of duplicate URLs present from 6a.
   d. Identify the number of duplicate URLs present from 6b.
   e. Deliver your updated crawl statistics from each repository that you crawl.

8. **(EXTRA CREDIT)** Dump the crawled data out of your Nutch content and then run [https://github.com/chrismattmann/tika-img-similarity/](https://github.com/chrismattmann/tika-img-similarity/) over it.
   a. Take the resultant JSON output and visualize it using D3 per the instructions in the Github repo.
   b. Are there any interesting clusters?
   c. Can you explain the clusters that you see?
   d. Develop a pull request for the Github repo to generalize it into [https://github.com/chrismattmann/etllib/](https://github.com/chrismattmann/etllib/)

4. Assignment Setup

4.1 Group Formation

You can work on this assignment in groups sized 4-5. Gouthami, the lead grader, should have sent you instructions on how to formulate groups. If you have questions contact:

Gouthami Kondakindi
[kondakin@usc.edu](mailto:kondakin@usc.edu)
Komal Dhawan
[komaldha@usc.edu](mailto:komaldha@usc.edu)
Kaustubh Vishwas Phatak
[kphatak@usc.edu](mailto:kphatak@usc.edu)

Use subject: CS 572: Team Details
4.2 Seed URLs for ACADIS, AMD and ADE

The seed URLs for the three repositories are:

NASA AMD:
http://gcmd.gsfc.nasa.gov/KeywordSearch/Home.do?Portal=amd&MetadataType=0

NSIDC ADE:
http://nsidc.org/acadis/search/

NSF ACADIS:
https://www.aoncadis.org/home.htm

You should begin collecting data as soon as you can since you will likely have to recrawl many times to get this right.

4.3 Downloading Apache Nutch

To get started with Apache Nutch, grab the trunk version, available from:
http://svn.apache.org/repos/asf/nutch/trunk
http://github.com/apache/nutch/

The latest version of Nutch that we will use in this assignment is Nutch 1.10-trunk.
Build Nutch with Apache Ant (available on most Unix systems) and Ivy by first checking out the code using a Subversion client:

svn co http://svn.apache.org/repos/asf/nutch/trunk nutch

or via Github:

git clone http://github.com/apache/nutch.git

Then enter the nutch directory and type:

ant runtime

Then wait a while. After Nutch is done building, you will have a runtime/local directory. That will be the Nutch deployment. The directory looks like this:

bin conf lib logs plugins test

4.4 Downloading and Installing Apache Tika

The quickest and best way to get Apache Tika up and running on your machine is to grab the tika-app.jar from: http://tika.apache.org/download.html. You should obtain a jar file called tika-app-1.7.jar. This jar contains all of the necessary dependencies to get up and running with Tika by calling it your Java program.
Documentation is available on the Apache Tika webpage at http://tika.apache.org/. API documentation can be found at http://tika.apache.org/1.7/api/.

You can also get more information about Tika by checking out the book written by Professor Mattmann called “Tika in Action”, available from: http://manning.com/mattmann/.

4.5 Some hints and helpful URLs

You may want to look into the Nutchpy library from Continuum Analytics as a method for generating crawling statistics from your Nutch databases and for your program to classify MIME types (task 2b):

https://github.com/ContinuumIO/nutchpy

A discussion of using Nutch to crawl dark data behind Ajax is ongoing and relevant to the Selenium task 4:

https://wiki.apache.org/nutch/AdvancedAjaxInteraction

Since you will be writing Nutch URL filters, you may want to have a look at the class APIdoc:

http://nutch.apache.org/apidocs/apidocs-1.9/org/apache/nutch/net/class-use/URLFilter.html

If you do the extra credit, you will need to look at D3:

http://d3js.org/

5. Report

Write a short 4 page report describing your observations, i.e. what you noticed about the dataset as you answered the questions in Part #3. Why do you think there were duplicates? Were they easy to detect? Describe your algorithms for deduplication. How did you arrive at it? What worked about it? What didn’t? Describe the URLs that worked and why? What MIME types did you retrieve from these websites? Were they mostly web pages? Did you get science data (e.g., HDF and NetCDF and Grib files?) If not, why? Did the Selenium plugin help? Did you get more data after installing the Tika updates and recrawling? Do you think you achieved good coverage of the 3 repositories?

Also include your thoughts about Apache Nutch and Apache Tika – what was easy about using them? What wasn’t?

6. Submission Guidelines

This assignment is to be submitted electronically, by 12pm PT on the specified due date, via Gmail csci572spring2015@gmail.com. Use the subject line: CSCI 572: Mattmann: Spring 2015: Nutch Homework: <Your Lastname>: <Your Firstname>. So if your name was Lord Voldemort, you would submit an email to csci572spring2015@gmail.com with the subject
Please note only one submission per team.

- All source code is expected to be commented, to compile, and to run. You should have (at least) one Java source file for each URLLFilter plugin you build in 7a and 7b and should also include other java source files that you added, if any. Do not submit *.class files. We will compile your program from submitted source.

- Include your program and/or script for generating MIME stats from a Nutch database. You may use Python and/or Java to generate this.

- Deliver your Nutch configuration e.g., your nutch-default.xml, nutch-site.xml, urlfilter-regex.txt file and any other configuration necessary to reproduce your results.

- Teams will be asked if they would like to contribute their crawled dataset to our NSF polar Amazon machine. This would include your Nutch segment data for each repository. If you want to do this, please identify in your report that you would like to do this, and send a message to the professor, to the TAs, and to the graders.

- Also prepare a readme.txt containing any notes you’d like to submit.

- Do not include tika-app-1.7.jar in your submission. We already have this.

- However, if you have used any external libraries other than Tika, you should include those jar files in your submission, and include in your readme.txt a detailed explanation of how to use these libraries when compiling and executing your program.

- Save your report as a PDF file (Lastname_Firstname_NUTCH.pdf) and include it in your submission.

- Compress all of the above into a single zip archive and name it according to the following filename convention:

  `<lastname> <firstname> _CSCI572_HW_NUTCH.zip`

  Use only standard zip format. Do not use other formats such as zipx, rar, ace, etc.

**Important Note:**

- Make sure that you have attached the file the when submitting. Failure to do so will be treated as non-submission.

- Successful submission will be indicated in the assignment’s submission history. We advise that you check to verify the timestamp, download and double check your zip file for good measure.

- Again, please note, only one submission per team. Designate someone to submit.

### 6.1 Late Assignment Policy

- -10% if submitted within the first 24 hours
- -15% for each additional 24 hours or part thereof