A Politics Mash-Up

Inki Hong (ikhong@seas.upenn.edu)

Faculty Advisor:
Ben Taskar (taskar@cis.upenn.edu)

Abstract

A mashup in the worldwide web is an application that collects data from multiple sources of information and presents them in a unique way. This paper describes a mashup on politics that integrates both geographical and
chronological dimensions regarding the 2008 presidential election of the United States of America.

Users of this mashup can browse through news articles, poll results, donations/contributions, and so forth arranged geographically over a time frame. It helps users of the system to organize diverse pieces of information, and facilitates geographical/chronological interpretation which is not apparent and hidden behind the disorganized scene.

**Related Work**

Most of the large market share portal sites have made some form of election section. Yahoo! has opened a democratic candidate mashup ([http://debates.news.yahoo.com/](http://debates.news.yahoo.com/)) which focuses on video interviews and poll results. CNN ([http://www.cnn.com/ELECTION/2008/](http://www.cnn.com/ELECTION/2008/)) has a broad coverage of election-related news. Although these sites put together mixed form of contents such as video and message board, they are still more of traditional portal sites than mashups in the way that these sites themselves are large producers of contents.

Some smaller sites such as 186kbps.com have a politics website (2008) that integrates a large variety of political news sources such as Reuters, USA Today and Yahoo! News. Only few of the articles, however, are groups together as meaningful categories. Mashing up is used to collect more than one source but the website itself doesn't add additional value presenting them. One attempt to
link politics on Google Maps service can be found in a web post (Sinreich, 2007) and a website by Matthew Kane (2005) which tries to track political campaign contributions geographically.

This project focuses on both geographical and chronological interpretations. A work by Veronis (2007) discusses correlation between # of candidate citations and the actual result (contrasted to poll results and the outcome.) Users are able to add geographical perspective to this kind of interpretation and prediction with the help of this mashup.

The New York Times has compiled up-to-date geographical campaign donation distribution on its web site where users can browse by individual candidates and further by time frames. It also lists past and upcoming schedules for each candidate by locations (The New York Times Company, 2008). An excellent site on campaign contributions is “The FundRace 2008” by Huffington Post (http://fundrace.huffingtonpost.com/) which provides its data to other sites as well.

This system also includes geographical/chronological media exposure based on the observation by Veronis (2007) that it has a significant correlation with the actual election. Very similar to the approaches that this system takes is “Map the Candidates” (Slate.com, 2008). Difference is that while this web site primarily focuses on candidate events, this system focuses on, based on the observation by Veronis (2007), candidates’ media exposure, poll results, contributions, of which one is expected to be very indicative of actual result and the other to be less forecasting than many people have relied on traditionally.
Technical Approach and Challenges

System Composition

The system is composed of three major subsystems. First, the back subsystem is responsible for collecting data from multiple sources of information, analyzing the assembled data, and extracting useful information from them. Second, the middle subsystem is responsible for permanently storing constructed information in a structured way so that it serves as a link between the front and back. Finally the front subsystem is responsible for presenting these pieces of information in a geographically and chronologically meaningful way.

Back End: Crawling

The back end is mainly based on several web crawlers which periodically fetches news articles, poll results, etc. that are available on the web and store them. Stand-alone web crawler programs were developed in Java language, version 6, to leverage a rich set of available libraries that facilitates development process. The Eclipse platform was used as it is the most widely used Java development tool. It was developed and tested in a regular desktop IBM-PC. For the networking modules, Sun Microsystems's own network libraries as well as Apache.org's HTTP Client (http://jakarta.apache.org/httpcomponents/) library have been used. The crawler is a single-threaded process.
The system only deals with HTML files, which are most of the time small in size, data are processed as memory-based InputStream-like objects rather than disk-based File-like objects. On processing HTML, it is converted to XML documents to make easy use of its tree structure ([http://java-source.net/open-source/html-parsers/jtidy](http://java-source.net/open-source/html-parsers/jtidy)). Once it has an XML representation, content extractors apply pre-specified rules and information retrieval techniques to extract useful information and store them in the middle subsystem.

**Back End: Web Sites**

PresidentPolls ([http://www.presidentpolls2008.com/](http://www.presidentpolls2008.com/)) is an excellent resource for poll results about presidential candidates. It has a nice structure where each result for the day is summarized in a table and all previous results can be downloaded by following single link.

All donation/contribution information can be searched at the Federal Election Commission ([http://www.fec.gov/](http://www.fec.gov/)). It also has parser-friendly structure only with larger amount of data.

News articles are, as one can naturally expect, very heterogeneous. Since it was not feasible to come up with a master parser for appropriate link detection, representative newspapers with high circulation were selected (BurrellsLuce, 2007). Other methods attempted to collect news articles is by meta-searching search engines. For example, USA Today ([http://www.usatoday.com](http://www.usatoday.com)), the newspaper with highest circulation in the United States, provides archive search service but only with headline and excerpt for free. In that case, a two-pass
Crawling was attempted. First pass collects headlines from the archive. In the second pass, those headlines are searched against popular search engines such as Yahoo! or Google to find links to the original URL.

As the number of web sites that is visited gets higher, time to build a new parser for the specific site should be reduced. Although every site has its own presentation structure, some information is extracted from a generalized parser using information retrieval heuristics. Examples are article title and keyword information which are often embedded in HTML meta tags and date/time information which has well-established pattern no matter where it appears. For possible future enhancement, article title information can also be deduced by heuristics from that in most pages title is displayed in largest font size. For date/time matching, reference materials for string pattern recognition will be consulted (Duda, Hart, & Stork, 2000). As another example, Gupta et al. (2005) introduces an approach of HTML content extraction based on heuristics and training where websites' categories are considered.

For all data the system collects, tagging presidential candidate name reference and location references for each piece of data is crucial. From the nature of data it crawls, most of the time name references will be of their last names properly capitalized, which makes it relatively simple for the content analyzer. Location recognition can be more complicated as explained above. Furthermore, machine processing of non-text, human-friendly resources without necessary annotation embedded within or nearby them is infeasible and alternative data source should be searched. Thus this project focused on extracting location information often stated as the first phrase along with date the article was written, as this is the case for many newspapers.
Back End: Geocoding

One important infrastructure for this project was the concept of geocoding, that is, converting city names and zip codes to latitude-longitude coordinates. This data integration was crucial because almost all of data we encounter on daily life are expressed in address as long as location is concerned whereas, although Google Maps provides full support for addresses, geography has to be expressed in terms of latitudes and longitudes for fast processing.

Thus this mapping data was also treated as information and was stored in the middle module for fast lookup. There are several sources that provide geocoding services. Web sites that provide Map APIs such as Google (2008) and Yahoo (2008) are examples.
Middle Subsystem

The middle subsystem provides permanent storage for the data. Since simple disk-based files are flat and require extra work to implement a tabular structure on top of them, two high-level systems were considered. A relational database has been standard choice for storing tabular structures but is very complex and has weaknesses in text searching.

<Locality-to-coordinate data>
Apache Lucene (http://lucene.apache.org/), an open-source search engine system, was chosen over traditional RDBMS for several reasons. First, since it is 100% Java, it is easier to integrate with other parts of the system. It's also lightweight and easy to use and to maintain indexes (tables) one of the reasons being that they are intuitively implemented as directories on file systems. An excellent tutorial by two of its developers (Gospodnetic & Hatcher, 2004) was enough to implement storing and searching. Finally, as the very reason of its existence, search engines maintain inverted tables very efficiently so that text-based searching is considerably faster than databases.

**Front End**

The front end is a web application based on the Google Maps service and Java Server Pages. Dynamic content creation is handled by JSPs and geographic components within it are presented by the Google Maps.

With little background on web applications, there were lots of technologies to learn. Both client-side JavaScript which Google Maps uses and server-side JSP which bridges client and server had many new concepts and were difficult to develop particularly because debugging was harder than stand-alone programs.
Browsing donations by geography
Conclusion

This system downloads pieces of information related to the 2008 Presidential Election from multiple websites. Then it organizes them by discarding unnecessary parts and compiling them in a structured format through indexing so that searching over the data is fast and accurate. Finally it presents meaningful information using web application with chronological and geographic dimensions.

Of the three subcomponents, the middle part was relatively easiest. Once I learned and was accustomed to Lucene search engine library putting in data as well as getting them back was well handled by the engine. As for data collection, although with the help of Apache library retrieving web pages was straightforward, each web site had its own way of composing data that incurred a lot of customizing work. Since I had little background on web applications, both client and server side scripting was a difficult task, on top of which relative lack of helper tools was added.
References

http://www.burrellesluce.com/top100/2007_Top_100List.pdf

Wiley-Interscience.


http://code.google.com/apis/maps/documentation/services.html#Geocoding


http://186kps.com/mashup/politics.php

http://www.slate.com/features/mapthecandidates/
