Web Search, Renmin University of China

Lab 2: non-programmers

Pierre Senellart (pierre@senellart.com)

14 July 2011

Labs can be made individually or by groups of two. Make sure to send by email to pierre@senellart.com, either at the end of the lab session or at the latest at midnight of the same day, an informal report about what you did in the lab, and the results you obtained. You do not have to finish all exercises, if you advance reasonably during the lab session but do not go to the end of the assignment, you will still get a passing grade.

1 Build Your Own Search Engine

We want to construct a specialized search engine that can retrieve online course materials on a variety of topics. Instead of implementing our own, we are going to use Google's Custom Search service.

1. Come up with a list of a few "important" Web sites that propose online course materials on a variety of topics. Try using Google Directory: http://www.google.com/Top/ for that purpose. Justify how you have chosen your list of sites.
2. Create a new custom search engine by going to http://www.google.com/cse/. You will need to create a Google account if you do not have one already. Fill in the required information and test your search engine.
3. Copy-paste the HTML code snippet that you obtain at the end in a text editor (e.g., Windows's Notepad). Save the result as search.html. Open this HTML document in a browser to test your custom search engine.
4. Add "Refinements" to your custom search engine (at http://www.google.com/cse/panel). Refinements are labels that you apply to sites to categorize them, or that automatically add terms to your search query. They can be selected by a user of the search engine to refine query results. Refinements can correspond to general areas for your courses (e.g., Computer Science, Humanities, Physics…). Sites can be labeled using the "Sites" configuration page. Test your custom search engine with refinements.
5. Explore the other options offered by Google Custom Search. Come up with examples of use of such a custom search engine.

2 Statistics about Web Sites

Alexa is a Web company that gathers statistics about Web sited. Most of these statistics come from a toolbar that users can install on their machines and sends information about user's habits to Alexa. In particular, Alexa maintains a list of the most visited Web sites, per country. Using http://www.alexa.com/, answer the following questions:

1. What is the most popular (in terms of its Web site) of the major universities in Beijing?
2. Apart from the homepage, what are the most visited Web pages of the RUC Web site? Why?
3. Compare the top sites visited in China and in the world. What are the most striking differences? What are the similarities?
4. Classify the 25 most visited Chinese Web sites according to their type (search engine, news, social network, etc.). Do you notice anything?
5. Compare the demographics (age, gender…) of users of Baidu, Google.co.hk, Google.com, Bing, Yahoo! Do you notice anything?

3 Small Worlds

1. Go to `http://www.google.com/`. By only following hyperlinks (and not running any query), try going to the homepage of RUC. How many steps do you need? Note down the path followed in the Web graph.
2. Go to `http://en.wikipedia.org/`. Click on "Random article". By only following hyperlinks from Wikipedia article to Wikipedia article, try going to the Wikipedia article on RUC. How many steps do you need? Note down the path followed in the Wikipedia graph.
3. The author of the XKCD Web comic noted the following: "Wikipedia trivia: if you take any article, click on the first link in the article text not in parentheses or italics, and then repeat, you will eventually end up at 'Philosophy'." (`http://xkcd.com/903/`). Verify this. What does it tell you about the graph structure of Wikipedia?

4 Mashups with Yahoo! Pipes

Mashups are Web applications that integrate and combine data from multiple Web sources to present them in a new way to a user. This exercise shows how to construct mashup applications using the Yahoo! Pipes graphical user interface. The goal will be to present information about news events, each event being accompanied by its localization displayed on a map. For that purpose, we integrate three sources of information:

1. A Web feed about current events in the world, in RSS format (e.g., The New York Times's world stories at `http://feeds.nytimes.com/nyt/rss/World`). Any such RSS feed is fine, though English is preferable to ensure precision of the geolocalization.
2. A geolocalization service. We use information from the GeoNames[1] geographical database, and specifically their RSS to GeoRSS converter, whose API is described at `http://www.geonames.org/rss-to-georss-converter.html`.
3. A mapping service. We use Yahoo! Maps[2].

Yahoo! Pipes[3] allows creating simple mashup applications (simply called *pipe*) using a graphical interface based on the construction of a pipeline of boxes connected to each other, each box performing a given operation (fetching information, annotating it, reorganizing it, etc.) until the final output of the pipeline. It can be used by non-programmers, though defining complex mashups still requires skill and experience with the platform.

[1] `http://www.geonames.org/`
[2] `http://maps.yahoo.com/`
[3] `http://pipes.yahoo.com/`

The mashup we want to build is demonstrated at `http://pipes.yahoo.com/webdam/geolocalized_news`: it asks the user for a feed URL, and displays with markers on a map the result of the geolocalization of each news item.

1. Go to the Yahoo! Pipes website and either log in using an existing Yahoo! account or create a free account. Once you follow the links for creating a pipe, you will be presented with the interface of the graphical editor: on the left, a list of all boxes that can be used inside a pipe; in the center, the workspace where you can build your pipe; in the bottom part, a debugger shows the output of the currently selected box.

2. Drag a "Fetch Feed" box on the workspace. Enter the URL in the box and connect it to the "Pipe Output" box at the bottom of the workspace by dragging a link from the output port of the initial box (shown as a circle on its bottom border) to the input port of the final box. Save your pipe and click on "Run pipe…" to see the result.

3. We are going to add some geolocalization information by using the "Location Extractor" operator of Yahoo! Pipes, which should be put in the middle of the two existing boxes. Save and run the pipe.

4. The location extractor of Yahoo! Pipes is not always as precise or complete as GeoNames. Study the documentation of the RSS to GeoRSS converter REST API. Use this API by trying to form URLs directly in your browser until you fully understand how it works. Then integrate it into your pipe by using a "URL Builder" whose output port is connected to the *url* parameter of the existing "Fetch Feed" box. Compare the results to what you had before.

5. To give a final touch to your pipe, add a "URL input" box to ask the user for the URL of the feed to be geolocalized. Save and test.

You can decide to publish your pipe to give other users access to it; if you want to keep playing with Yahoo! Pipes, you can try enriching your pipe by retrieving data from multiple RSS feeds, using a Web search operator to discover feeds dealing with a given topic, adding to feed items images obtained by querying Flickr with keywords from the description of the item, and so on. You can also look at the vast library of published pipes to get some inspiration.