Extending Yioop! with Geographical Location Local Search

Committee Members
Dr. Chris Pollett
Dr. Soon Tee Tech
Dr. Mark Stamp

By
Vijaya Sinha

CS298 Writing Project Defense
Agenda

- Motivation
- Project goal
- Background
- Planet.osm and hostip.info
- Yioop!
- Modifications to Yioop!
- Test and Results
- Demo
- Conclusion
Motivation

- Commercial search engines like Google and Yahoo! provide location based searches
- Source of Geographic data is copyright protected and not available easily
Project goal

- Extend Yioop! an open source search engine with location based search in an Open source Alternative
- Provide local searches on the search results.
- Plot results on the map to aid lookup.
What is required?

- Geographic data with spatial information
- We use planet.osm data
- User’s location to provide local searches
- We use hostip.info database
Free geographic data of the whole planet

Sources of data from GPS devices, aerial photography or local knowledge

Comes in different file formats namely PBF (efficient binary), compressed OSM XML and o5m.

We use OSM XML format to build the Yioop! iterator to go over OSM data and index it
<?xml version="1.0" encoding="UTF-8"?>
<osm version="0.6" generator="OpenStreetMap planet.c"
timestamp="2011-02-16T01:11:04Z">

    <node id="270387" lat="50.8777604" lon="-1.5338646"
timestamp="2006-08-31T14:39:25Z" version="1"
changeset="99256" user="nickw" uid="94">
    <tag k="created_by" v="osmeditor2" />
    <tag k="name" v="Jacklin &amp; Escuela"/>
        <tag k="operator" v="VTA"/>
    <tag k="route_ref" v="46;47;66"/>
    </node>

    <way id="33289926" user="Roozbeh" uid="6069" visible="true"
version="3" changeset="597814" timestamp="2009-04-16T21:52:32Z">
    <nd ref="378341727"/>
    <nd ref="330146871"/>
    <tag k="created_by" v="Potlatch 0.10f"/>
        <tag k="landuse" v="residential"/>
    <tag k="name" v="Mobilodge of Milpitas"/>
    </way>

</osm>
Identifying points of interest

- Data set is large
- We need to identify some points of interest that would make meaningful search results
- Index only nodes and ways that are named
Node example

<node id="270387" lat="50.8777604" lon="-1.5338646" timestamp="2006-08-31T14:39:25Z" version="1" changeset="99256" user="nickw" uid="94">
<tag k="created_by" v="osmeditor2" />
<tag k="name" v="Jacklin & Escuela"/>
  <tag k="operator" v="VTA"/>
<tag k="route_ref" v="46;47;66"/>
</node>
Way example

<way id="33289926" user="Roozbeh" uid="6069" visible="true" version="3"
   changeset="597814" timestamp="2009-04-16T21:52:32Z">
   <nd ref="378341727"/>
   <nd ref="330146871"/>
   <nd ref="330146872"/>
   <nd ref="330146873"/>
   <nd ref="330146874"/>
   <nd ref="378341727"/>
   <tag k="created_by" v="Potlatch 0.10f"/>
   <tag k="landuse" v="residential"/>
   <tag k="name" v="Mobilodge of Milpitas"/>
</way>
Hostip.info

- User’s location can be obtained using PHP superglobal variable 
  \$_SERVER[‘REMOTE_ADDR’]
- Need to convert ip address to geo–location.
- We use hostip.info database
- Hostip.info can be used to convert between ip–address and geo–location.
IP LookUp

Please enter the ip address to look up:
130.65.11.68

Go

State :California
Country :UNITED STATES
Country_code :US
City :San Jose, CA
Latitude :37.304
Longitude : -121.85
Yioop!

- Yioop! is an open source search engine written in PHP developed by Dr. Chris Pollett.
- The main components of Yioop! are queue server and fetcher.
- The queue server is the coordinator of the crawls and send URLs to the fetcher to download.
- Fetcher downloads the pages, extracts summaries of pages and builds a partial index.
Yioop! allows crawls of different file formats like Mediawiki xml and ODP RDF

The data to crawl is stored in timestamped folder on the queue server

Using the same concept we build an archive iterator just for Osm data
Osm_archive_iterator

- We read the osm data from the timestamp folder
- We start reading node and make html pages of the nodes with title as name the page and description consisting of nodeid’s latitude, longitude and other textual data
- Similarly for ways, we make html pages with the name as title, wayid’s and other text data as description.
We make html pages for node and way points as Yioop! comes with the some built in processors like html, image, pdf, doc processors.

Converting the osm data into html pages would help us make use of html processor to index pages as html pages.
After indexing node page

<html>
<head>
<title>Jacklin &amp; Escuela</title>
</head>
<body>
<h1>nodeid -270387 lat 50.8777604 lon -1.5338646 operator vta</h1>
</body>
</html>
After indexing way page.

<html>
<head>
<title> Mobilodge of Milpitas </title>
</head>
<body>
<h1> wayid 33289926  nodeid 378341727 nodeid 330146871 nodeid 330146872 nodeid378341727 landuse residential created_by Potlatch 0.10f </h1>
</body>
</html>
Modifications to Yioop!

- Modifications to model.php – Base class for all models in the search engine. Modified to make latitude, longitude information part of ways.
- Displayresults_helper a helper that helps to automate the task of certain tags was modified so that garbage data was not part of the page summary.
- Search view was modified to include a div to display maps.
- A new javascript file map.js was added
Query Results: (Calculated in 0.783887 seconds. Showing results 0 - 10 of 17)

**Caltrain**
rail B11 Santa Clara, CA Union Pacific Railroad
http://www.yahoo.com/nodeid/wayid-119420087 Rank: 1.00 Rel: 10.91 Prox: 4.00 Score 11.4 [Cached] [Similar]
[Inlinks]
[Show map]

**Caltrain**
rail B11 Santa Clara, CA Union Pacific Railroad
Calculated using:

```php
function calcDist($lat_A, $long_A, $lat_B, $long_B) {
    
    $distance = sin(deg2rad($lat_A))
    * sin(deg2rad($lat_B))
    + cos(deg2rad($lat_A))
    * cos(deg2rad($lat_B))
    * cos(deg2rad($long_A - $long_B));

    $distance = (rad2deg(acos($distance))) * 69.09;
    $actualscoredis = log(1/$distance+1);
    return $actualscoredis;
}
```
Distance calculation..

- The score calculated is added to the total score so the documents are ranked automatically by their distance.
- The search results are ranked according to the distance.
Test and results

- Precision—measure of how many documents returned to a given query are relevant.

\[
\text{Precision} = \frac{|\text{Rel} \cap \text{Res}|}{|\text{Res}|}
\]

- Recall—measure of how many of all known documents are retrieved by the system.

\[
\text{Recall} = \frac{|\text{Rel} \cap \text{Res}|}{|\text{Rel}|}
\]
Nominatim

- Nominatim is a tool that is used to search osm data set by name and address and powers the home page of openstreetmap.org.

- Nominatim indexes named (or numbered) features with the OSM data set and a subset of other unnamed features (pubs, hotels, churches, etc)
Recall comparison with Nominatim
Precision - comparison with Nominatim
Recall comparison with Yahoo! Local and Ask city
Precision comparison with Yahoo! Local and Ask city
Conclusion

- We were able to get local search results and rank them according to user’s location.
- The results are mapped on a map for easy lookup.
- To better performance, bounding box filter can be used so searches are faster.
Demo...
Questions!!