Jeocrowd - Collaborative Searching of User-Generated Point Datasets

George Lamprianidis
Institute for the Management of Information Systems
G. Mpakou 17
11526 Athens, Greece
glampr@imis.athena-innovation.gr

Dieter Pfoser
Institute for the Management of Information Systems
G. Mpakou 17
11526 Athens, Greece
pfoser@imis.athena-innovation.gr

ABSTRACT

Geospatial data has become an important resource in today's Web applications not only as type of content, but also as metadata. Despite its undisputed usefulness, issues need to be addressed with respect to the availability, the accuracy, and the cost of the data. The advent of Web2.0 created several creative-commons initiatives addressing geospatial dataset creation and countless (mobile) applications have been producing large amounts of point cloud datasets. In this work, we demonstrate how to query user-contributed point-cloud data using a collaborative Web-based approach.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data Mining

Keywords

data mining, geospatial data fusion, user-contributed content, mapreduce

1. INTRODUCTION

An increasing number of contexts and applications use space as the primary means to structure and access information simply because geospatial reasoning ("where" is easier to comprehend than "what") is essential to everyday problem solving. The boost in the number of applications is coupled with an increasing demand for sizable and ever-up-to-date geospatial data. Here, with the proliferation of the Internet as the primary medium for data publishing and information exchange, we have seen an explosion in the amount of online content available on the Web. Prominent examples of such User-Generated Content (UGC) for the geospatial domain include (i) photo sharing sites such as Flickr, Panoramio, Picasa, and many others, and (ii) microblogging sites with a geotagging feature such as Twitter, Facebook, Google+ as well as related photo sharing sites (twitpic).

While such geodata streams use geographic reference systems and are thus comparable at the coordinate level, they do pose a challenge with respect to semantic integration, i.e., to which spatial object, or, Point of Interest (POI) does the geotag in question relate, and location ambiguity. Consider here the example of geotagged Flickr photos. While many of them might refer to the same POI, the recorded coordinate information hardly matches up due to positioning inaccuracy, or, simply, wrong geotagging. For example, for large scale objects different users will record different coordinates. Some of them might even capture the POI from a distance, which will set the coordinates completely off.

The objective of this work is to transform available UG geocontent into meaningful chunks of information, geospatial datasets, obtained with simplicity and speed comparable to that of Web-based search. To achieve this task, we propose a search method utilizing crowdsourcing concepts implemented as a Web-based, collaborative search tool termed *Jeocrowd*[1].

In recent years various approaches have been developed that aim at exploiting UGC and here especially tag information to derive geospatial data, or, in more general terms, place information. The following discussion should give an indication as to what has been achieved and makes no claim of completeness. [1] gives an approach to derive geospatial shape information from geotagged Flickr photos. Yahoo! GeoPlanet WOEIDs as part of the metadata are exploited by means of direct access to the Flickr database. [4] extract place semantics from Flickr tag information based on the spatial distribution of the data. In [3], again Flickr data is used to identify places and also relationships among them (e.g., containment) based on tag analysis and spatial clustering. While all the above approaches have certain similarities with the basic premise of our work, i.e., to extract geospatial information from user-contributed datasets, our contribution will be towards defining a collaborative data mining framework that actively involves the user and its resources in the process.

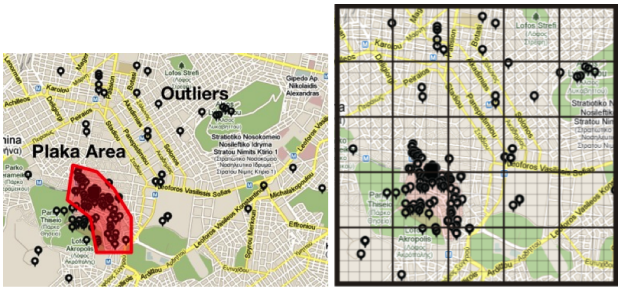
The outline of the remainder of this work is as follows. Section 2 surveys the type of user-generated data we will exploit in this work. Section 3 describes the advocated computational approach, while the actual computing framework is given in Section 4. Section 5 describes the resulting application and its interface as it will be demoed at the conference. Finally, Section 6 presents conclusions and directions for future work.

2. USER-GENERATED AND GEOSPATIAL DATA

Users generate data by means of many different applications in which the geospatial aspect does not play a major role but is simply used to index and access the collected data. As mentioned, prominent examples here are photo sharing sites and microblogging services using geotagging features. To illustrate the potential for

[1]The tool is named Jeocrowd, which is a combination of Geo + Crowd, using a J instead of G, to emphasize the use of Javascript as its core calculation platform.

ACM SIGSPATIAL GIS '11 November 1-4, 2011. Chicago, IL, USA



(a) Point cloud (b) Grid approach

Figure 1: Point cloud for the "Plaka" area in Athens, Greece.

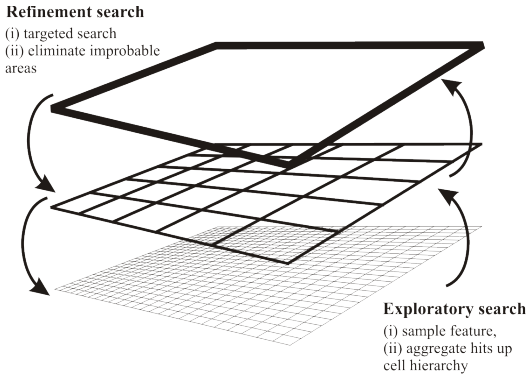


Figure 2: Exploratory and refinement search traversing up and down the hierarchical grid structure

geospatial data generation, consider the use of flickr data for the computation of feature shapes of various spatial objects including city scale, countries and other colloquial areas [1]. The approach is based on computing primary shapes for point clouds that are grouped together by Yahoo! GeoPlanet WOEIDs (Yahoo! Where On Earth IDs), which are part of the flickr metadata. Flickr currently contains 150+ million geotagged photos.

Querying the API with a specific search term returns in essence a "point cloud" referring to a specific POI. In other words, the geotagging of each photo represents an independent observation of the "true" location of the POI. The ambition of this work is to take point clouds as input and to identify an approach for extracting geospatial location information from it. Consider the example point cloud in Figure 1(a), which shows geotags of retrieved flick images for the query "Plaka". The result includes also locations outside the actual area (shaded polygon). The task at hand is how to derive the area information from this point cloud data. Complicating things is the fact that the location of a POI might be that of a point or area feature. Prominent examples here are "Central Park" (area feature) and "Parthenon" (point feature) [2].

Overall, to extract place information from point clouds, we propose a method based on *online* (live search), *collaborative* (many users querying the same term speed up the search), *Web-based* (browser-based computing) *crowdsourcing* (users contributed expertise - the search term, and resources - computation power and bandwith). The core contributions of this work are (i) a collaborative spatial search implemented by means of (ii) Web and browser-based computing utilizing a modified MapReduce approach as described in the following two sections.

3. COLLABORATIVE SPATIAL SEARCH

The basic task at hand is how to derive spatial datasets from user-contributed point cloud data that have basically no quality guarantee. A typical approach would entail the clustering of the points (cf. [1]) in combination with outlier detection. The latter should eliminate points that have been assigned false coordinate information in connection with image labels.

In Jeocrowd, we employ a grid-based clustering approach (cf. [5]). Point cloud locations are aggregated by means of a 5×5 hierarchical grid. The grid at the basic level (Level 0) has a spacing of 50m $\times$ 50m, at Level 1 the grid has a spacing of 250m, etc. When querying point cloud APIs, we record for each grid cell a count, i.e., how many photos are located in a specific cell. The search entails "intelligently" filling the cells of the grid with point cloud data based on a specific search term, e.g., "Plaka, Athens". Figure 1(b) shows the example of "Plaka" with a hierarchical grid overlay. Using various thresholds such as occupancy and connectedness, the result will comprise as set of connected cells whose spatial extent will be the position of the searched for spatial feature (POI). The typical result will be a polygon, with the accuracy of the shape being limited by the spacing of the grid.

Using a grid has the advantage that the search (i) can be parallelized, i.e., cells can be populated simultanesouly from various data sources and (ii) is in fact continuous, i.e., as additional user-generated data becomes available, the search result will be refined. For example, city boundaries may change in subsequent searches to reflect newly observed data.

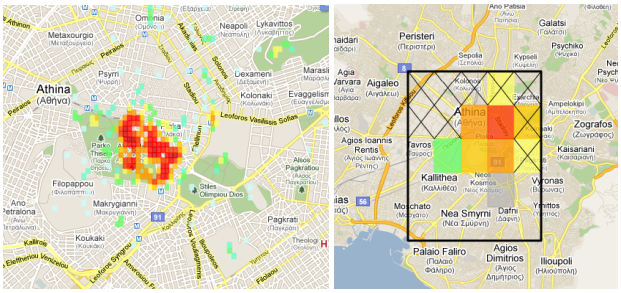
Since dealing with remote data sources, our goal is to retrieve as little data as possible so as to speed up search and reduce traffic. To this effect we split our search into an exploratory part, which tries to quickly find a good estimate of the extent of the spatial feature and a refinement part, which then tries to explore the shape of the spatial feature in greater detail. As shown in Figure 2, the hierarchical grid is used to expand the search area after exploratory search and eliminate unlikely areas during the refinement stage.

Exploratory search aims at randomly retrieving points to get an overview of the spatial extend of the spatial feature, i.e., is it a point location (statue, coffee shop) or are we looking for an area feature (neighborhood, city, country). Point cloud, and specifically in this work, Flickr data is retrieved in chronological order starting with the oldest photo. The percentage of how much data is retrieved in this step is a search parameter. Figure 3(a) shows the result of the exploratory search for "Plaka" (Level 0). Specifically a neighborhood map is shown, in which colors from blue to red are assigned to cells based on how many neighboring cells have also collected point cloud data. Isolated cells are colored blue and surrounded cells are shaded red.

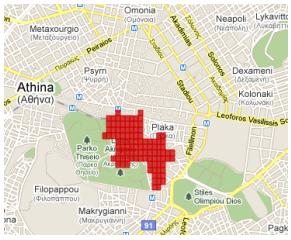
During the *refinement*, the search is forced into areas that are suspected of being part of the feature, but for which we have not retrieved (many) points in the exploratory search. This is due to the fact that users tend to take disproportionally many pictures of the most popular locations. Here, the refining search uses the search term in connection with location estimates to retrieve additional photos for this "unpopular areas".

The refinement search is based on occupied cells of the hierarchical grid. The number of points associated with each cell as a result of the exploratory search are aggregated to higher levels. The aggregation stops once fewer than a minimum number of cells at a given level are occupied. The refinement search then proceeds with search-

[2]The authors acknowledge however that whether a location can be represent by a point or an area depends on the scale and (practical) degree of abstraction at which the data is represented.



(a) Exploratory search (b) Refinement search



(c) Result

Figure 3: Searching for Plaka, Athens

ing these super cells. Figure 3(b) shows the super cell of "Plaka". Colored cells have been discovered during exploratory search. Cells marked with an X have not been searched yet. The refinement step searches all cells that are either empty or have few hits. Newly discovered points are added and the respective counts and neighborhood information is updated at all levels of the hierarchical grid.

The search examines all cells at a level and eliminates as many as possible. Eliminated cells will not be examined further at a lower level. Cells are eliminated if they do not attract a minimum number of point cloud locations despite forcing a search in this area. Using a hierarchical grid, the goal is to rule out cells as early as possible, i.e., at high levels, since examining larger areas at lower levels is costly (many small cells). The result of the "Plaka" search after completing the refinement step is shown in Figure 3(c).

Our approach can also detect multi locations, if the search term refers to more than one spatial entities. For example, the search for "Olympic Stadium, Athens" will return two locations, the Marble Stadium and the new Olympic Stadium.

4. BROWSER-BASED MAPREDUCE

To facilitate collaborative search, one not only needs to design a search strategy and respective data structures, but also a computing framework that allows users to perform this task simultaneously. Here, we propose Browser-based computing in connection with the MapReduce [2] concept. We introduce a Web application with the ability to apply a computational model such as MapReduce so as to speed up the procedure by utilizing the users' machines to perform the computations and, thus, offloading the server.

As we saw in Section 3, during the exploratory search, the retrieved photos are in ascending order of the timestamp of the date taken. During the refinement search the photos are ordered in terms of latitude and then longitude. This ordering permits the *search task to be split into multiple subtasks*, each of which can be assigned a non-overlapping range of pages to be retrieved from the Flickr API and to be handled by the above search procedure independently on separate browsers (parallelizing the search) (cf. Figure 4). To keep track of assigned and completed subtasks, we keep on the server for each search term one search record and multiple result records

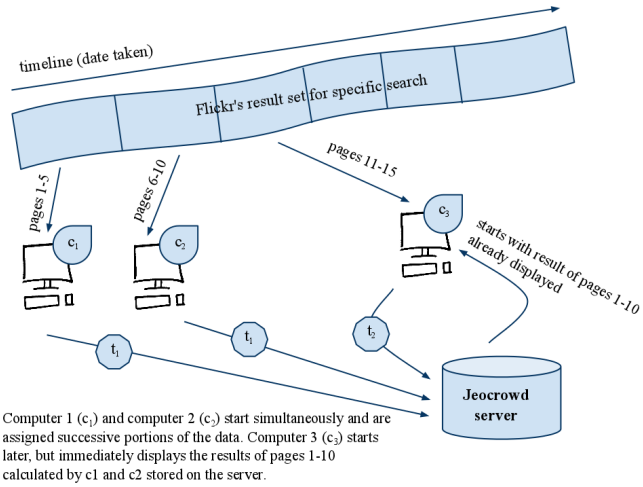


Figure 4: Browser-based MapReduce in Jeocrowd

marking the exact page ranges that were assigned and completed by clients (browsers). Hence, when a new request for the same search term is received, we send to the client the current state of the search and the next range of pages currently missing. For each subtask, the results are collected and merged with already existing ones in the search record.

To persist the state of the search for each query, we keep all the information needed to reconstruct the lowest (most detailed) grid plus some counters that record the progress of the search process. Having that, it is a pure computational task to reconstruct higher levels and resume the search. In the current implementation, each grid is stored as a collection of grid cells and we save for each its coordinates and the list of photos found in its enclosed area.

The difference to the original MapReduce idea is that after the completion of a subtask by a client, the respective result record is merged with the corresponding search record immediately. This allows us to communicate the current state of the search to all connecting clients. To prevent the search from missing results due to clients failure to report back (browser closed or crashed, network connection interrupted, etc.) the system imposes a *timeout* to the clients in order to complete the retrieval, computation and reporting of the result. This allows for scheduling a resubmission of a task to another client.

One constraint to the performance is that due to dependencies in our algorithm, all exploratory search subtasks for a search term have to be completed before performing a refinement search. Further, in its current implementation, the refinement search propagates down cell levels. Here, all searches for a specific level have to be completed before progressing to the level below. This is because after each level is completed, isolated cells for that level are discarded, and so are enclosed cells in lower levels. If a task were to start processing the lower level before the elimination process had occur, it might have started querying a cell that would eventually be discarded! Hence, if all the subtasks for a specific search step have been assigned, but not all completed, a new subtask will be created for an already assigned portion of the search and we will consider the result of the first respective finished subtask. Overall, these techniques are very common in MapReduce implementations.

In the Jeocrowd approach, the server is only used for coordinating computation tasks and storing results. All computation takes place in the browser. This has many advantages, the main one being the increase in computational capacity. The basic work flow of the search

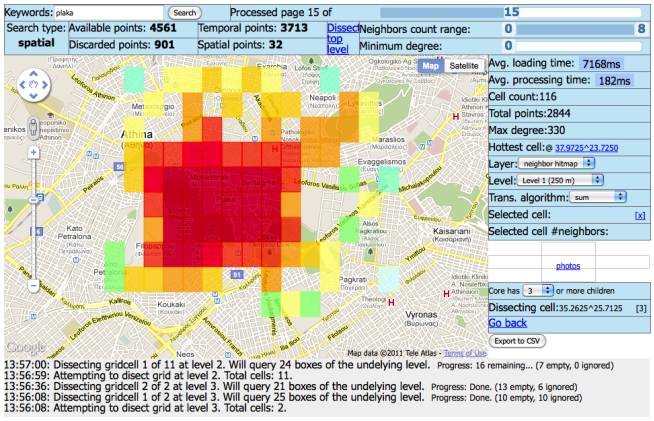


Figure 5: Jeocrowd search interface

is shown in Figure 4.

Moving the computation part from the server to the browser means that the communication with the external sources (e.g., Flickr in the case of the Jeocrowd demo) will have to be initiated by the client as well. This introduces a challenge insofar since Javascript executed in the browser needs to contact the external server and then use the data retrieved for computation and visualization. This contradicts with the *same domain policy* enforced by browsers according to which a script running on a page is allowed to access resources from another source only if the protocol, the host, and the port match. To overcome this problem we use JSONP (JSON with Padding). What this does is that it wraps (pads) the data returned from the external source inside a function call that eventually gets executed in the browser. For this to work, (i) the function needs to be already defined in the Javascript code, and (ii) the external source must agree to send the response data wrapped in that same function name. Hence the security threat is eliminated and the function makes sure to deliver the data to the Javascript workspace. Using this approach, the server is relieved of the great bandwidth constraint that typically Web application dealing with third-party data APIs are burdened with.

5. JEOCROWD DEMO

The scope of the demo will be to showcase the Jeocrowd search interface as shown in Figure 5. The Web interface provides (i) information regarding the search status and (ii) the basic search parameters. The dominant component is the map, which visualizes the search results. Three sliders control various parameters. The top slider allows the user to change the number of pages that will be fetched in this particular request, i.e., the number of flickr images. This number can be anything from 2 (we need at least one to get the total number of pages) up to the last page of the result set. The two sliders below affect on what is displayed in the map. One slider sets the neighborhood threshold for cells shown, i.e., a cell can have 0 to 8 surrounding neighbor cells. The other sets the threshold for the minimum number of photos per cell, i.e., show only the cells that attracted more than 10 photos. The search statistics (to the left of the sliders and above the map) include (i) the total number of available photos for the specific search term, (ii) the number of photos downloaded for exploratory (temporal) and refinement (spatial) search, and (iii) the number of discarded photos. The information column to the right of the map contains additional information regarding the search, such as average loading and processing time for sets of retrieved images (pages), the total number of grid cells that appear on the map, the total number of photos in all the grid cells, as well as the number of photos in the grid cell that attracted the most photos. One can further zoom to the "hottest" cell, i.e., the cell with the most photos. Next, we have some controls that allow the user to change the way the results are displayed. The first one changes the coloring of the map by switching from a *hit map* (cells are all the same color, but the opacity of the cell is analog to the number of photos it attracted) to a *neighborhood heat map* (cells have different colors depending on the number of their neighbors, following the convention that red cells have the most neighbors while blue ones have none). The interface also allows one to retrieve information for a specific cell such as visualizing all photos and captions retrieved. At the very bottom of the interface is a log, which tracks the progress of the search, keeping some stats with respect to each step. This log can be exported to a CSV file and easily imported to external spreadsheet tools for further analysis.

6. CONCLUSIONS

Jeocrowd provides an online, collaborative, browser-based spatial search tool that employs crowdsourcing techniques to harness user-contributed point-cloud data by means of user-contributed expertise and computing resources. The Jeocrowd application will be freely available to the users before and after the demo sessions to query arbitrary shaped spatial objects ranging from smaller point features ("Riesenrad, Vienna") up to city-scale area features ("Berlin"). Any popular enough place name should yield good results.

Directions for future work are to improve the performance of the algorithm, to include further data sources, and to consider existing geospatial metadata for inferring relationships and hierarchies between spatial objects. The latter will enable us to merge results for searches that refer to the same POI but did not use exactly the same keywords, as well as refine searches for large areas by fusing results from places that are known to be "part-of" it.

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme - Marie Curie Actions, Initial Training Network GEOCROWD (http://www.geocrowd.eu) under grant agreement No. FP7-PEOPLE-2010-ITN-264994.

7. REFERENCES

[1] A. Cope. The Shape of Alpha. Where 2.0 conf. presentation, http://where2conf.com/where2009/public/schedule/detail/7212, 2009.

[2] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. In *Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation - Volume 6*, pages 10–10, Berkeley, CA, USA, 2004. USENIX Association.

[3] S. Intagorn and K. Lerman. Harvesting geospatial knowledge from social metadata. *Knowledge Creation Diffusion Utilization*, (May):1–10, 2010.

[4] T. Rattenbury and M. Naaman. Methods for extracting place semantics from flickr tags. *ACM Trans. Web*, 3:1:1–1:30, January 2009.

[5] W. Wang, J. Yang, and R. R. Muntz. Sting: A statistical information grid approach to spatial data mining. In *Proceedings of the 23rd International Conference on Very Large Data Bases*, VLDB '97, pages 186–195, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.