Clustering and Visualizing Geographic Data Using Geo-tree

Che-An Lu, Chin-Hui Chen, and Pu-Jen Cheng
Department of Computer Science and Information Engineering
National Taiwan University
Taipei 106, Taiwan
{jast.lu, dongogo}@gmail.com, pjcheng@csie.ntu.edu.tw

***Abstract*—Plotting lots of geographical data points usually clutters up a map. In this paper, we propose an approach to provide a summary view of geographical data by efficiently clustering. We present a novel data structure, called Geo-tree, which is extended from quadtree, and then develop two algorithms, which use Geo-tree to cluster geographical data and visualize the clusters with a heatmap-like representation. The experimental results show that our approach is very efficient in a large scale, compared to K-means and HAC, and very helpful for browsing.**

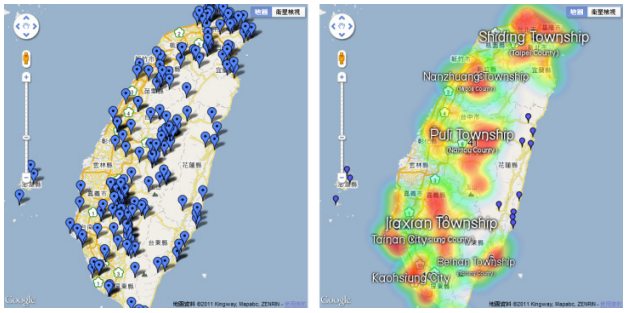
Keywords-Geo-tree; clustering; visualization; geographic data

I. INTRODUCTION

More and more search results nowadays are suitable to be displayed with web mapping systems. Compared to returning a linear ranking list, displaying geographic data on maps can provide useful spatial information about locations for users. It also can help users browse (zoom in/out or pan) the 2D data to explore further messages. Plotting markers on the map is one of the easiest ways to display geographic data (Figure 1(a)); however, it causes too many markers overlapping with each other, and the annotations on the map are covered by those markers as well. Therefore, some clutter reduction techniques have been proposed.

Reference [5] categorizes clutter reduction techniques into three main types: appearance, spatial distortion, and temporal. Among these three types, appearance (e.g. clustering and filtering) is the most appropriate to be used in de-cluttering geographic data [4]. Hence several clustering techniques have been introduced.

As a map often supports multi-level resolutions for browsing, some applies hierarchical clustering to map presentation. A recent paper [4] proposes an HAC-like (Hierarchical Aggregation Clustering) structure to cluster geographic data first. When browsing partial data, it grouped them by simply selecting corresponding nodes in the structure. Such browsing process is very fast, because no real clustering computation, i.e., HAC, is required during browsing. The time complexity for selecting nodes is nearly constant-order. However, the clustering results will not change while a user pans to different locations. In other words, the clustering results are static because they already exist before users' operation. Others dynamically cluster the partial data a user is going to browse [1, 10].



(a) (b)

Figure 1: Both images visualize the geographic data about "Natural Scenic spot in Taiwan". (a) is the result of plotting all points on the map, and (b) is our result.

They directly apply some clustering algorithms with linear-order complexity for each browsing operation. In this way, the clustering results will vary according to different one's browsing focuses. Nevertheless, the cost in time is too high to be used in real applications.

After clustering geographic data, visualizing these clustering results is another research issue. Reference [4] proposes a visualization method using Voronoi polygons. However, its cost in time is too high to be applied in an interactive system (e.g. more than 2000 seconds for rendering 5000 points). Other solutions such as using icons [2], cells [7], bounding boxes, or convex hull [3] to represent clusters also appear. But the drawback of these solutions is that users cannot see the data's distribution within a cluster. On the contrary, paper [8, 9] focuses on visualizing geographic tendency without considers the latent clusters of data.

In this paper, clustering and visualization algorithms based on quadtree [6] are proposed to alleviate the above problems. We modify quadtree into a so-called Geo-tree, which help users efficiently browse geographic data on a map (Figure 1(b)). Since users can zoom to different levels and pan to different locations, our clustering approach based on Geo-tree should be dynamic for different zoom-levels and locations. Moreover, in order to interact with users, the approach should respond in acceptable time. For visualizing the clusters, we design an algorithm to render a heatmap-like image which can help users catch the distribution within clusters. Moreover, we also design a scoring function to select representative geographic labels to enhance the location information for users.

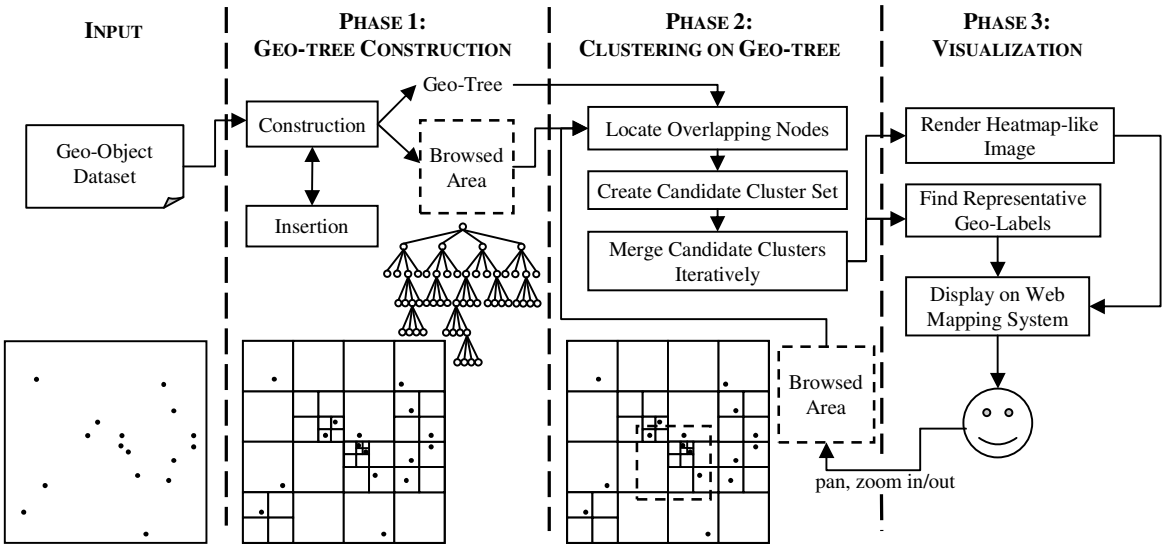


Figure 2: System architecture

II. PROPOSED GEO-TREE SYSTEM

The input of our system is geographic data. In general, we annotate a geographic data as $\mathbb{O} \triangleq \{o\}$, which contains a set of geo-objects. Each geo-object o is a tuple (θ, x, y) containing an unique object ID θ and its coordinate (x, y), which can be obtained from latitude and longitude using Mercator projection. The output is a summarized view of geo-objects according to user's browsing area. The system architecture is illustrated in Figure 2. It is composed by three main phases: Geo-tree Construction, Clustering on Geo-tree and Visualization.

In this section, we introduce what is Geo-tree first, and then state the details of each phase.

A. Geo-Tree

Geo-tree splits a map into tiles until all geo-objects are in different tiles. Each split divides the current tile into four equal area tiles. The main difference between quadtree and Geo-tree is that Geo-tree maintains extra indexing information for all pairs of adjacent nodes. Hence, it contains the characteristics listed below:

- Each node corresponds to a rectangle on the map.
- Nodes in the same tree-level (depth of Geo-tree) have equal area.
- A node's parent has four times the area of itself.
- Each node has links to its geographically adjacent nodes, including adjacent leaf nodes and adjacent nodes in the same tree-level.
- Each node maintains the number and the centroid of geo-objects it contains.

Therefore, a node in Geo-tree is defined as $N = (r, l, p, c, L_{siblings}, L_{leaves}, x, y, n)$. The meaning of each element in N is described in Table I.

TABLE I. THE DESCRIPTION OF ELEMENTS IN NODE N.

Notation	Description
r	the bounding rectangle of N
l	the tree-level of N
p	the parent of N
c	the four children of N
$L_{siblings}$	the links to adjacent nodes in the same tree-level
L_{leaves}	the links to adjacent leaf nodes
x, y	the centroid (the center of gravity) of geo-objects bounded by r
n	the number of geo-objects bounded by r

B. Construction Phase

We insert geo-objects one by one to construct a Geo-tree. The main purpose of insertion is to find a corresponding leaf node for each geo-object. If a leaf node contains more than one geo-object, it will be split into four sub-nodes. However, in order to restrain the space needed for Geo-tree, we also use a threshold $l_{\max}$ to indicate the deepest permissible tree-level. Otherwise, two geo-objects in the same location will cause infinite loop of split.

Notice that larger $l_{\max}$ indicates that we prefer less objects stay in the same node, and that we need a larger space for saving the tree structure (a deeper tree structure).

C. Clustering Phase

There are three main steps to cluster geographic data based on Geo-tree: (1) Locating nodes that overlap with a browsed area A, (2) Creating a set of candidate clusters, and (3) Merging candidate clusters until stopping conditions are satisfied.

First, given a browsed area A, we traverse Geo-tree from root to the first node N_s whose area is smaller than A. Then

we add it into a node list $\mathbb{L}$. For each new added node in $\mathbb{L}$, we locate its adjacent nodes through $L_{siblings}$, and add the sibling overlapping with the area A into $\mathbb{L}$. Repeat this step until no more nodes overlapping with A.

Second, for each node in $\mathbb{L}$, traverse its sub-tree and stop at those nodes whose l equals to l_{stop} or leaves. Then we insert these stopping nodes into a candidate cluster set $\mathbb{C}$.

Third, iteratively compute the distances between adjacent clusters in $\mathbb{C}$ (the adjacent clusters can be found through $L_{siblings}$ and L_{leaves}). Then merge the closest pair of clusters (here we use Euclidean distance between two clusters' centroids to be the distance metric). Repeat the merging step until all clusters are far enough from each other (farer than a distance threshold).

Notice that there are two key factors making our clustering approach efficient: (1) the clustering targets are the nodes (tiles) in the candidate cluster set $\mathbb{C}$ rather than geo-objects, and (2) we only need to compute the distance between adjacent clusters rather than pair-wise distance between clusters.

D. Visualization Phase

If the density of geo-objects in a cluster is lower than a threshold, we will directly plot these points on the map. Otherwise, we render a heatmap-like image for this cluster and also tag the cluster with several representative geo-labels.

1) Rendering heatmap-like image

To render a heatmap-like image, we first use the bounding rectangles r to decide the size of the image. Then aggregate the energy of each pixel, which can be obtained from each tile in the cluster. Here, we use the two-dimensional Gaussian function to model the energy spread from a tile centered at (x, y):

$$f(x', y') = ne^{-\left(\frac{(x'-x)^2}{2\alpha_x{}^2}+\frac{(y'-y)^2}{2\alpha_y{}^2}\right)}$$

where n is the number of geo-objects contained in the tile, and α_x and α_y are the width energy will spread. Hence, if n is larger or pixel (x', y') is closer to (x, y), the pixel will get higher energy. Finally, we use the aggregated energy to create a heatmap-like image, which will then be displayed on the map.

2) Finding representative geo-labels

Here we define a geo-label as a location containing ID, name, parent's ID, latitude, longitude and area (e.g. $\langle 3, \text{Taipei City}, 2, 25.091, 121.559, 271.799\rangle$).

We filter out the geo-labels that are too far away from cluster's centroid. For the remaining geo-labels, we compute a representative score S to decide their importance to the cluster:

$$S = \frac{2UV}{U+V}, U = \frac{A_o}{A_l}, V = \frac{A_o}{A_c}$$

where A_l is the area of the label, A_c is the area of the cluster, and A_o is the approximate overlapping area of the label and the cluster. When U is higher, the label is more accurate for the cluster; when V is higher, the label can cover more region of the cluster. The concepts of these equations are similar to Precision, Recall and F1-measure. Then we use a threshold to decide which labels are qualified for helping users understand the geographic position of the cluster. In our system, users can arbitrarily choose to display these representative geo-labels or not.

III. EXPERIMENTS

We evaluated our system in two aspects, efficiency and accuracy. For efficiency, we computed the time for Geo-tree construction among different scale of geo-objects ranged from 1K to 25K. We also compared the time for clustering in different scale with two well-known clustering algorithms, K-means and HAC. Both of the compared algorithms are still widely applied in many researches nowadays. In the accuracy part, we evaluated the clustering quality with three standard measures: purity, entropy and NMI (normalized mutual information).

A. Dataset

The dataset used in this paper contains 29,497 geo-objects located in Taiwan. Each geo-object contains the following information: id, name, geographic type, address, latitude and longitude. There are totally 51 geographic types in this dataset such as university, gas station, parking lot, post office, hotel, etc. With the address information in each geo-object, we can identify the object's located district (e.g. Da-an District, Taipei). Hence, we can search our dataset with query like "night markets in Taipei". Also notice that each district has a corresponding entry in our geo-label database. The database contains 395 geo-labels in Taiwan. Since the dataset is from real data, we assume that the objects' distribution of a query (e.g. hotels in Taipei) is similar to a map-based search result. The dataset can also be any other geographic data such as photos with latitude and longitude, or micro blogs with location (e.g. twitter, plurk through devices with GPS).

B. Efficiency

All the experiments in this section were executed on a 2.33 GHz Intel Core 2 Duo with 8GB of RAM.

1) Construction

We evaluated the time for Geo-tree construction among different number of geo-objects. In each scale, we randomly sampled five different set of data and report the average time. The result is shown in Table II(A). Notice that the time for construction is linear to the number of geo-objects, and it takes less than 0.025 seconds for 25K geo-objects.

Table III: The results of measures computed in fifteen data with different approaches (P stands for purity, E stands for entropy, and N stands for NMI).

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	avg
Geo-tree	**P**	**0.99**	**0.90**	0.93	0.91	**1.00**	0.91	**0.89**	0.80	**0.85**	**0.98**	0.79	0.71	**0.94**	0.84	**0.93**	**0.89**
	E	**0.01**	0.10	0.13	0.09	**0.00**	0.13	0.16	0.22	**0.16**	**0.06**	0.37	0.26	**0.12**	0.32	**0.20**	**0.16**
	N	0.94	**0.92**	**0.91**	0.93	0.83	**0.83**	**0.86**	0.83	0.83	**0.88**	0.67	0.84	**0.82**	0.63	0.72	0.83
HAC	**P**	**0.99**	**0.90**	**0.94**	**1.00**	**1.00**	0.91	**0.89**	**1.00**	0.85	0.69	0.79	**0.88**	0.71	**0.88**	**0.93**	**0.89**
	E	**0.01**	0.10	**0.12**	**0.00**	**0.00**	0.13	0.16	**0.00**	**0.16**	0.52	0.37	**0.13**	0.47	0.28	**0.20**	0.18
	N	**0.98**	**0.92**	**0.91**	**1.00**	**1.00**	**0.83**	**0.86**	**1.00**	**0.87**	0.55	0.67	**0.88**	0.56	0.73	**0.78**	**0.84**
K-means	**P**	**0.99**	**0.90**	0.93	0.81	0.78	**0.92**	**0.89**	0.80	0.82	0.95	**0.96**	0.69	0.91	0.87	0.70	0.86
	E	0.02	**0.08**	0.13	0.21	0.29	**0.12**	**0.13**	0.23	**0.16**	0.15	**0.08**	0.34	0.17	**0.27**	0.50	0.19
	N	0.96	0.90	0.80	0.66	0.67	0.79	0.81	0.70	0.82	0.84	**0.90**	0.69	0.75	**0.75**	0.27	0.75

TABLE II: (A) TIME FOR GEO-TREE CONSTRUCTION (B) TIME FOR CLUSTERING

# points	1k	5k	10k	15k	20k	25k
Construct	0.004s	0.032s	0.072s	0.122s	0.162s	0.202s

(A)

# points	1k	5k	10k	15k	20k	25k
HAC	0.302s	36.83s	293.7s	980.3s	2319s	4524s
K-means	**0.014s**	**0.110s**	0.248s	0.458s	0.505s	0.789s
Geo-tree	0.127s	0.217s	**0.244s**	**0.248s**	**0.249s**	**0.252s**

(B)

2) Clustering

We evaluated the time for clustering in 6 different data size comparing to HAC and K-means (see Table II(B)). It can be seen that the time for Geo-tree clustering comes to a static value while the data size grows. However, HAC and K-means grows with third power and linear to the size of data. It is because the number of tiles needing to be clustered in our approach is limited by the screen size. Therefore, even though the data size grows, the time for clustering will be bound in a static value.

C. Accuracy

We evaluated the quality of our clustering results, and compared our results with K-means and HAC. We totally generated 15 different sets of clusters to be the ground truth. Geo-objects in the same cluster came from the same district in our ground truth (ex: parking lots in Taipei City, elementary schools in Kaohsiung), and the districts between clusters were mutually exclusive. The number of clusters ranged from 3 to 6.

Then we evaluated the quality of clustering results using three different measures: Purity, Entropy and normalized mutual information (NMI). These were standard measures for evaluating cluster quality.

We set the number of clusters in K-means and HAC to the ground truth's, and automatically generated the number in our approach. The average numbers of clusters in the ground truth and in our approach were 4.13 and 4.6 respectively. The results of K-means, HAC and our system are shown in Table III. For most data, our overall performance is comparable to K-means and HAC.

IV. CONCLUSION

In this paper, we propose an approach to provide a summary view of geographic data by efficiently clustering. We present a novel data structure, called Geo-tree, and utilize it to cluster and visualize geographic data with heatmap-like representation. We evaluate the efficiency and accuracy of our approach. The experimental results show that our approach is very efficient in a large scale, compared to K-means and HAC, and our accuracy is comparable to theirs.

REFERENCES

[1] S. Ahern, M. Naaman, R. Nair, and J. H.-I. Yang. 2007. World explorer: visualizing aggregate data from unstructured text in geo-referenced collections. In *Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries* (JCDL '07). ACM, New York, NY, USA, 1-10.

[2] S. Burigat and L. Chittaro. 2005. Visualizing the results of interactive queries for geographic data on mobile devices. In *Proceedings of the 13th annual ACM international workshop on Geographic information systems* (GIS '05).

[3] M. Cristani, A. Perina, U. Castellani, and V. Murino. 2008. Content visualization and management of geo-located image databases. In *CHI '08 extended abstracts on Human factors in computing systems* (CHI EA '08).

[4] J.-Y. Delort. 2010. Hierarchical cluster visualization in web mapping systems. In *Proceedings of the 19th international conference on World wide web* (WWW '10).

[5] G. Ellis, A. Dix, "A Taxonomy of Clutter Reduction for Information Visualisation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1216-1223, Nov./Dec. 2007.

[6] R. A. Finkel and J. L. Bentley. Quad trees a data structure for retrieval on composite keys. *Acta Informatica*, 4(1), 1974.

[7] Girardin, F.; Calabrese, F.; Fiore, F.D.; Ratti, C.; Blat, J.; , "Digital Footprinting: Uncovering Tourists with User-Generated Content," *Pervasive Computing, IEEE* , vol.7, no.4, pp.36-43, Oct.-Dec. 2008, doi: 10.1109/MPRV.2008.71.

[8] Hotta, H.; Hagiwara, M.; , "A Neural-Network-Based Geographic Tendency Visualization," *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT '08.*

[9] Hotta, H.; Hagiwara, M.; , "Online Geovisualization with Fast Kernel Density Estimator," *Web Intelligence and Intelligent Agent Technology, 2009. WI-IAT '09.*

[10] A. Jaffe, M. Naaman, T. Tassa, and M. Davis. 2006. Generating summaries and visualization for large collections of geo-referenced photographs. In *Proceedings of the 8th ACM international workshop on Multimedia information retrieval* (MIR '06). ACM, New York, NY, USA, 89-98.