Efficient Indexing Structure for Scalable Processing of Spatial Alarms

Myungcheol Doo[◇] Ling Liu[◇] Nitya Narasimhan[♠] Venu Vasudevan[♠]

[◇] School of Computer Science, Georgia Institute of Technology, Atlanta, GA
[♠] Applied Research Center, Motorola, Schaumburg, IL
{mcdoo, lingliu}@cc.gatech.edu, {nitya, venu.vasudevan}motorola.com

ABSTRACT

We present the design and implementation of a new indexing technique, Mondrian tree. The Mondrian tree indexing method partitions the entire universe of discourse into spatial alarm monitoring regions and alarm-free regions. This enables us to reduce the number of on-demand alarm-free region computations, significant savings of both server load and client-to-server communication cost. We evaluate the efficiency of the Mondrian tree indexing approach and show that the Mondrian tree offers significant performance enhancements on spatial alarm processing at both the server side and the client side.

Categories and Subject Descriptors

H.2 [**Database Management**]: Database ApplicationSpatial Databases and GIS; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Indexing Methods*

General Terms

Algorithm

Keywords

Indexing, Spatial Alarms, Mobile Applications, Location-based Services

1. INTRODUCTION

Spatial alarms extend the idea of time-based alarms to the spatial dimension. They serve as personal reminders to the mobile users upon their arrival of a specified location of interest. An example of a spatial alarm is *Locale* [1] which enables one of the pre-defined cellphone ring settings upon arrival of future reference location of interest. For example, Alice sets a spatial alarm on her office in Georgia Tech campus. When she arrives on campus or within two miles of her office, *Locale* changes the ringer setting to the pre-defined setting such as silence.

ACM GIS'10 November 2-5, 2010. San Jose, CA, USA

2. SPATIAL ALARM PROCESSING

The naïve strategy for spatial alarm processing is periodic evaluation. High frequency is essential to ensure that few or none of the alarms are missed. Periodic evaluation, though simple, can be extremely inefficient due to frequent and often high rate of irrelevant alarm evaluations [4].

2.1 Grace Period

The grace perioid is to incorporate the spatial locality of the alarms and the motion behavior of mobile objects. Given a mobile object and an alarm, the grace period can be computed based on the distance between the current position of the user and the alarm monitoring region.

The concept of grace period has two performance implications. At the server side, we can skip the check of any spatial alarms owned or subscribed by p during the grace period regardless whether or not the location update events for p may occur during this period. At the mobile client side, mobile client can enter sleep mode for the spatial alarm evaluation application during the grace period.

2.2 Alarm-Free Region

Regardless of the total number spatial alarms subscribed or owned by a user, only those alarms that are in her close vicinity have non-zero probability of being fired. The idea of alarm free region (AFR) utilizes this spatial locality of alarms and the motion behavior of mobile clients. An AFR is defined as a rectangular region that does not contain any spatial alarms. As long as the mobile client moves within an AFR, the client is free from the need for alarm check, which significantly reduces the amount of unnecessary evaluations of spatial alarms at the alarm evaluation server. However, computing a rectangular AFR takes from $O(n)$ to $O(n \log^3 n)$, where n is the number of spatial alarms used in the AFR computation [7]. The worst case is that the client periodically update its location and the server computes the AFR on each location update.

3. MONDRIAN TREE INDEXING

3.1 Basic Ideas and Properties

The Mondrian indexing is a rectangular region partitioning method. It takes as an input the spatial alarms in the given universe of discourse and generates a region partition tree with two types of spatial regions as the output. A unique feature of the Mondrian tree is that it indexes not only spatial alarm monitoring regions but also the remaining regions, called **alarm-free regions (AFR)**, which do not

have any spatial alarms. Thus, each leaf node of Mondrian tree is either an alarm monitoring region or an AFR. The goal of such a design is to allow the spatial alarm evaluation engine to easily locate the current region of a mobile client, and more importantly, to evaluate a spatial alarm only when the mobile subscriber of the alarm is inside of the alarm monitoring region.

3.2 Partitioning the Entire Region

Figure 1 shows the steps of partitioning an area with one spatial alarm into 5 disjoint regions (one alarm monitoring region and four AFRs). The corresponding Mondrian trees for the four steps in Figure 1 are shown in Figure 2. At each step we choose a split dimension which is a perpendicular to one of the 2D coordinate axes. The split dimension may be chosen by alternating between x-axis and y-axis. In step 1, x-axis is selected as a splitting dimension. Now the entire region is divided into two smaller regions, R_1 and R_2. The root of the Mondrian tree, which covers the entire region, has two children, each of which points to R_1 and R_2 respectively as shown in Figure 2(a). In step 2, R_2 in Figure 1(a) is chosen to be further partitioned because it contains T_1. We alter the split dimension from x to y-axis and split R_2, which results in R_3 and R_4 as shown in Figure 1(b) and 2(b). We keep selecting a region and partitioning it until the remaining rectangle has the same region as the spatial alarm monitoring region as shown in Figure 1(d).

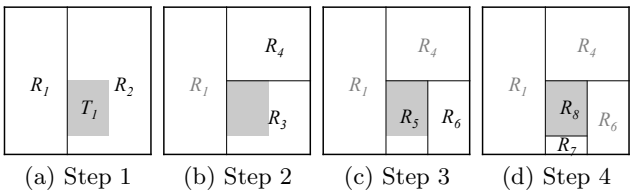


Figure 1: Step-by-step partitioning

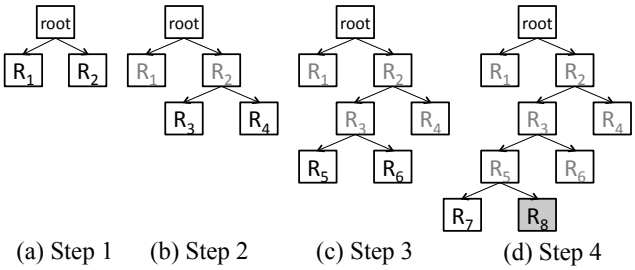


Figure 2: Step-by-step building a tree

3.3 Comparisons with R-tree and k-d tree.

Mondrian indexing partitions the entire universe of discourse while other multi-dimensional indexing algorithms are interested in regions that have queries or objects of interests. Figure 3 and 4 illustrates some basic differences that distinguish Mondrian tree from R-tree and k-d tree in the context of spatial alarm processing. The user's current location is labeled as a star.

Mondrian tree returns R_9 as the AFR for this user as shown in Figure 3(a). In contrast, R-tree or k-d tree do not return anything. Therefore we need to locate the spatial alarms nearby a mobile client and compute the AFR.

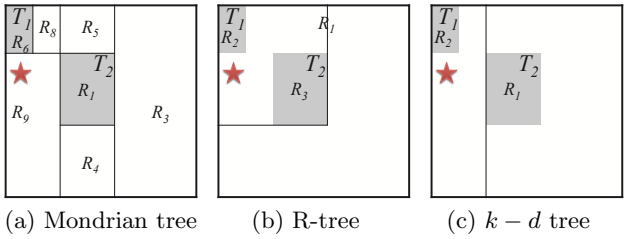


Figure 3: Comparison of partitioning scheme

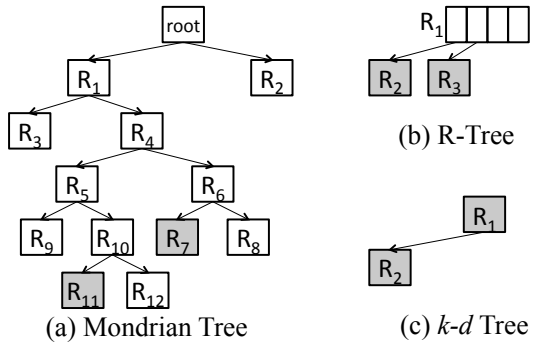


Figure 4: Comparison of building trees

3.4 Search AFRs and Spatial Alarms

The search for mobile user's current region in Mondrian tree is similar to the search algorithm of binary tree. From the root it compares the position of the mobile object with the left and the right children's' region to determine if the search should proceed in the left subtree or the right subtree of the index. Figure 5 shows how to find the leaf node according to the current location.

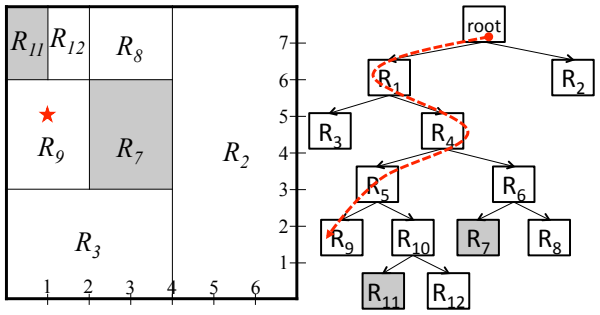


Figure 5: A search path to the current node

4. EXPERIMENTAL EVALUATION

4.1 Experiment Setup

All the experiments presented in this paper are conducted by extending the GTMobiSim mobility simulator [6]. The

simulator generates a set of traces of moving vehicles on a real world road network. Maps are obtained from the National Mapping Division of the U.S. Geological Survey[3] in the form of Spatial Data Transfer Format[2]. We use the map of Metro Atlanta, which covers an area larger than 1,000 km^2. We initially place vehicles randomly on the road segments according to traffic densities determined from the traffic volume data [5]. The generated traces are simulating vehicles moving on the roads of metro Atlanta for a period of ten minutes. At each intersection, a vehicle will choose a direction and a road segment to travel randomly from the set of available choices. The number of vehicles generated varies from 2,000 to 10,000. The default number of vehicles is 2,000. Also the number of spatial alarms varies from 2,000 to 10,000. The default number of alarms is 10,000.

We evaluate the performance of the Mondrian tree index under the server-centric architecture with periodic evaluation approach (MPRD) and grace period approach (MGP), the distributed architecture with periodic evaluation (M+PRD) and grace period approach (M+GP). In the distributed architecture, there are n Mondrian trees that indexe only spatial alarms that each user owns. Therefore, the search time for AFR or an alarm is reduced compared to server-centric architecture. We compare Mondrian tree algorithms with Grid index (GPRD) and R-tree index (RPRD) using periodic evaluation and dynamic AFR computation at the server.

The main factor affecting the processing of spatial alarms using Grid indexing is the size of grid cell. The smaller cells the Grid has, the more cells each alarm will intersect with. The larger cells the Grid has, the more alarms will be within a single cell, which increases the server-side evaluation cost. To provide a fair comparison with Grid index against R-tree and Mondrian tree, we choose the size of the cell as 2,730m $\times$ 2,730m in the experiments reported in this section.

4.2 Client-side Performance Evaluation

Figure 6(a) shows the average number of wake-up for each mobile client. In all approaches except MGP and M+GP, clients have the same number of wake-up because clients periodically wake up wait for server to inform them if they are in an AFR or an alarm monitoring region. These results show that periodic evaluation has two orders of magnitude larger number of wakeups compared to Mondrian grace period approach (M+GP) because clients in M+GP wake up only when their grace period expires. The average size of AFR decreases as the number of alarms increases. Clients staying in the smaller AFR has shorter grace periods. Therefore, the clients in MGP wake up more frequently as the number of alarms increases.

Similar results happen to the number of crossing AFRs shown in Figure 6(b). In M+PRD and M+GP, each client only creates and maintains its own Mondrian tree, which has only the number of alarms that the client subscribes to, and thus very small compared to the centralized server-side Mondrian tree (MPRD). Therefore the average AFR size is much bigger for distributed Mondrian tree approach and has the lower number of crossing AFRs.

In summary, when using periodic evaluation approaches, clients wake up periodically and wait for the server to inform them if they enter an alarm monitoring region or send them the new AFR if they cross the existing AFR. In contrast, with grace period approach the spatial alarm application on the client side enters the sleep-mode longer than periodic approaches.

Next set of experiments on the client side aims at measuring energy efficiency. We compute the power consumption of client device for CPU usage and network usage. According to [8] even in idle mode in which there is no data transfer via a wireless network device, the network device consumes power, and the activity for data transfer via network devices consumes more power than the computing activity. Due to the server processing time as shown in Figure 6(e), clients in GPRD and RPRD wait longer and more frequently for AFR information from the server and thus consume more battery power. This is evident from Figure 6(d). Clients in M+GP consumes the least power and wakeups the least. M+PRD approach, relatively speaking, consumes less power than Grid, R-tree, and server-side (centralized) Mondrian tree approaches. However, the M+PRD approach consumes more power than M+GP because M+GP requires clients to wake up only when their grace period is expired instead of periodically as in M+PRD. RPRD and MPRD approaches consume most power. In MPRD, clients cross their AFR frequently due to the small size of AFRs. In RPRD, clients have to wait longer due to the longer response time compared to the GPRD approach.

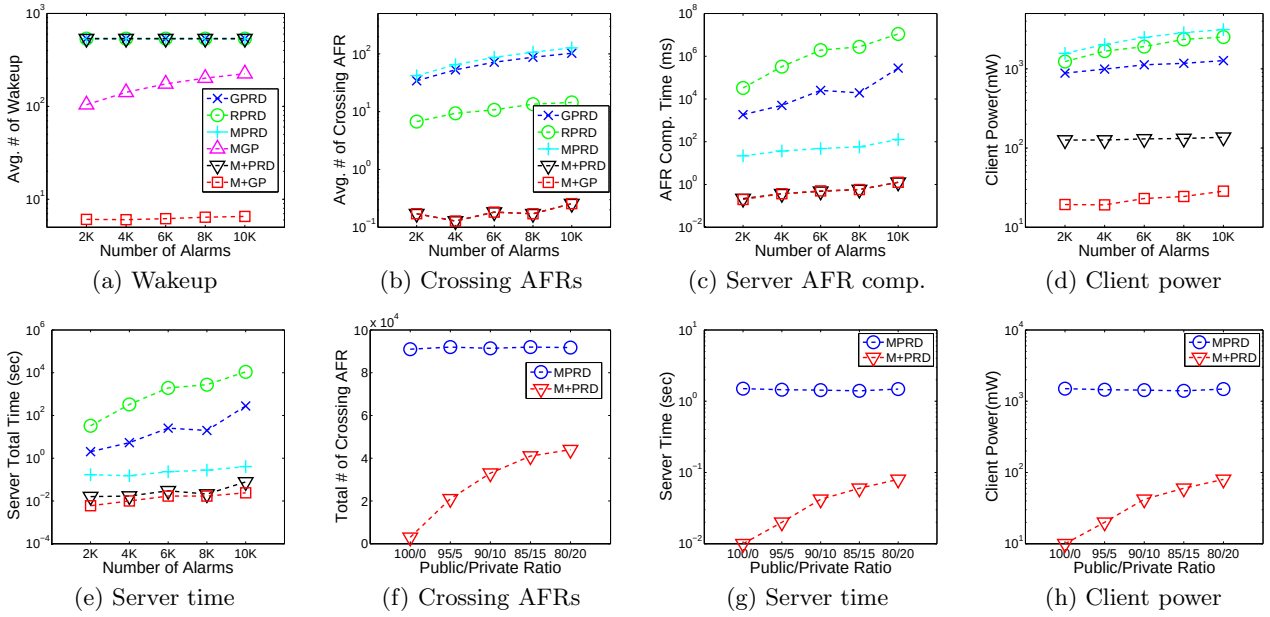
4.3 Performance Evaluation in Server-side

Mondrian tree approach indexes both spatial alarms and AFRs at the same time. Therefore, it does not require on-demand AFR computation as Grid index (GPRD) and R-tree index (RPRD) approaches. Instead the server needs to pay only AFR search time over the Mondrian tree. When the Mondrian tree is hosted at the server as a big tree, the search time is relatively high compared to the search time for distributed Mondrian index (M+GP). However, the search cost of MGP is closer to M+GP when comparing with Grid index and R-tree. GPRD and RPRD takes not only search time but also AFR computing time. RPRD and GPRD take much longer time in computing AFR compared to Mondrian algorithms (MPRD, M+PRD, M+GP) because Mondrian algorithms only pay the AFR search cost over the respective Mondrian tree (centralized or distributed Mondrian tree). Figure 6(c) shows server AFR computation time. RPRD takes more time than GPRD due to the higher search cost for finding the nearby alarms. Interesting to note is that distributed Mondrian tree outperforms the centralized Mondrian tree regardless whether we use grace period or periodic evaluation. This further demonstrates the efficiency of distributed Mondrian algorithms.

Figure 6(e) shows the total server time, which consists of computing AFRs (in R-tree and Grid index) or searching AFRs (in Mondrian tree) and processing spatial alarms. The more AFR crossings a client experiences, the more time the server spends in computing or searching AFR in addition to processes alarms. Given that distributed Mondrian tree approach (M+GP) has the largest AFR size on average, there is fewer AFR crossings experienced at the client side. Therefore, M+GP has the least server time for spatial alarm processing, while RPRD consumes the largest server time.

4.4 Performance Comparison of MGP and M+GP

In this section, we compare the performance of centralized Mondrian tree (MPRD) and distributed Mondrian trees (M+PRD) under different combination of private and public alarms. To test the extreme performance of distributed



(a) Wakeup (b) Crossing AFRs (c) Server AFR comp. (d) Client power

(e) Server time (f) Crossing AFRs (g) Server time (h) Client power

Figure 6: Comparison of Mondrian tree algorithm with R-tree and Grid index (a ∼ e) and Comparison of Mondrian Trees (f ∼ h)

Mondrian tree approach, we vary the percentage ratio of private and public alarms at 100:0, 95:5, 90:10, 85:15, 80:20, and measure the total server processing time, the client energy consumption, and the total AFR crossings. The number of users is 6,000 in this set of experiments. As the percentage of public alarms increases, we observe an increase in the total number of AFR crossings, the total server processing time (sec), and the client energy consumption (mW) in M+PRD compared to MPRD. This is because the distributed Mondrian indexing is more sensitive to the increase of public alarms. The more alarms a client has, the more frequently the client may cross the AFRs (Figure 6(f)), the higher load the server will have (Figure 6(g)), and more energy will be consumed at each mobile subscriber (Figure 6(h)). In comparison, for MPRD there is only one Mondrian tree in the server-side and the addition of k public alarms does not add more than k alarms to the total number of alarms in the system. Thus, the number of AFR crossing does not change significantly in comparison to M+PRD. This set of experiments shows that centralized Mondrian tree approach is more effective when there are a larger number of public alarms in the system, whereas the distributed Mondrian tree approach prevails over centralized Mondrian approach when the number of public alarms is relatively smaller.

5. CONCLUSIONS

We have described the design and development of the Mondrian tree index for efficient processing of spatial alarms. The main distinguishing feature of Mondrian tree compared with conventional spatial indexes such as R-tree family and Grid is that Mondrian tree approach indexes not only spatial alarms but also AFRs, enabling fast lookup of AFRs instead of on-demand computation of AFRs. Another novelty of the Mondrian indexing framework is its ability to utilize the characteristics of spatial alarms to create and maintain one Mondrian tree for each mobile subscriber, which is particularly effective when there is relatively small number of public alarms compared to the private and shared alarms in the system. Our experiments show that the distributed Mondrian tree (M+GP, M+PRD) can dramatically minimize the amount of unnecessary spatial alarm processing compared to centralized Mondrian tree (MPRD and MGP), R-tree, and Grid indexing structures with periodic evaluation and on-demand AFR computation.

Acknowledgement

This work is partially sponsored by grants from NSF NetSE and NSF CyberTrust, an IBM SUR grant, and a grant from Intel Research Council.

6. REFERENCES

[1] Locale. http://www.twofortyfouram.com/.
[2] Spatial data transfer format. `http://www.mcmcweb.er.usgs.gov/sdts/`.
[3] Us geological survey. `http://www.usgs.gov/`.
[4] B. Bamba, L. Liu, , A. Iyengar, and P. S. Yu. Distributed processing of spatial alarms: A safe region-based approach. ICDCS, 2009.
[5] B. Gedik and L. Liu. Location privacy in mobile systems: A personalized anonymization model. In *Proc. IEEE ICDCS*, 2005.
[6] P. Pesti, B. Bamba, M. Doo, L. Liu, B. Palanisamy, and M. Weber. Gtmobisim: A mobile trace generator for road networks. Technical report, 2009.
[7] S. Prabhakar, Y. Xia, D. V. Kalashnikov, W. G. Aref, and S. E. Hambrusch. Query indexing and velocity constrained indexing: Scalable techniques for continuous queries on moving objects. In *IEEE Transactions on Computers*, 2002.
[8] V. Raghunathan, T. Pering, R. Want, A. Nguyen, and P. Jensen. Experience with a low power wireless mobile computing platform. In *Proc. of International Symposium on Low power Electronics and Design*, 2004.