Scalable Processing of Spatial Alarms

Bhuvan Bamba[1⋆], Ling Liu[1], Philip S. Yu[2⋆⋆], Gong Zhang[1], and Myungcheol Doo[1]

[1] College of Computing, Georgia Institute of Technology
[2] Department of Computer Science, University of Illinois at Chicago
{bhuvan,lingliu,gzhang3,mcdoo}@cc.gatech.edu, psyu@cs.uic.com

Abstract. Spatial alarms can be modeled as location-based triggers which are fired whenever the subscriber enters the spatial region around the *location of interest* associated with the alarm. Alarm processing requires meeting two demanding objectives: high accuracy, which ensures zero or very low alarm misses, and system scalability, which requires highly efficient processing of spatial alarms. Existing techniques like periodic evaluation or continuous query-based approach, when applied to the spatial alarm processing problem, lead to unpredictable inaccuracy in alarm processing or unnecessarily high computational costs or both. In order to deal with these weaknesses, we introduce the concept of *safe period* to minimize the number of unnecessary spatial alarm evaluations, increasing the throughput and scalability of the server. Further, we develop alarm grouping techniques based on locality of the alarms and motion behavior of the mobile users, which reduce *safe period computation costs* at the server side. An evaluation of the scalability and accuracy of our approach using a road network simulator shows that the proposed approach offers significant performance enhancements for the alarm processing server.

1 Introduction

Time-based alarms are effective reminders of future events that have a definite time of occurrence associated with them. Just as time-based alarms are set to remind us of the arrival of a *future reference time point*, spatial alarms are set to remind us of the arrival of a *spatial location of interest*. Thus, spatial alarms can be modeled as location-based triggers which are fired whenever a mobile user enters the spatial region of the alarms. Spatial alarms provide critical capabilities for many location-based applications ranging from real time personal assistants, inventory tracking, to industrial safety warning systems.

A mobile user can define and install many spatial alarms; each alarm is typically shared by one or many other users. Alarms can be classified into three categories based on the publish-subscribe scope of the alarm as *private*, *shared* or *public* alarms. *Private* alarms are installed and subscribed to exclusively by the alarm owner. *Shared* alarms are installed by the alarm owner with a list of k ($k > 1$) authorized subscribers and the alarm owner is typically one of

⋆ This work was performed while the author was at IBM T.J. Watson Research Center
⋆⋆ This work was performed while the author was at IBM T.J. Watson Research Center

the subscribers. Mobile users may subscribe to *public* alarms by topic categories or keywords, such as *"traffic information on highway 85North"*, *"Top ranked local restaurants"*, to name a few. Each alarm is associated with an *alarm target* which specifies the location of interest to the user; a region surrounding the alarm target is defined as the *spatial alarm region.* The *alarm trigger* condition requires that subscribers of the alarm be notified as soon as they enter the spatial alarm region.

Processing of spatial alarms requires meeting two demanding objectives: high accuracy, which ensures no alarms are missed, and high scalability, which guarantees that alarm processing is efficient and scales to large number of spatial alarms and growing base of mobile users. The conventional approach to similar problems involves periodic evaluations at a high frequency. Each spatial alarm evaluation can be conducted by testing whether the user is entering the spatial region of the alarm. Though periodic evaluation is simple, it can be extremely inefficient due to frequent alarm evaluation and the high rate of irrelevant evaluations. This is especially true when the mobile user is traveling in a location that is distant from all her location triggers, or when all her alarms are set on spatial regions that are far apart from one another. Further, even a very high frequency of alarm evaluations may not guarantee that all alarms will be successfully triggered. The *spatial continuous query approach* would process a spatial alarm by transforming the alarm into a *user-centric* continuous spatial query. Given the alarm region of radius r around the alarm target and the mobile user's current location, the transformed spatial query is defined by the query range r with the mobile alarm subscriber as the focal object of the query. The query processor checks if the obtained query results contain the alarm target object. This process repeats periodically until the alarm target is included in the query results at some future time instant. The obvious drawback of this approach is the amount of unnecessary processing performed in terms of both the number of evaluations and the irrelevant query result computation at each evaluation. A more detailed discussion of the weaknesses can be found in our technical report [6].

Spatial alarms can be processed using server-based infrastructure or client-based architecture. A server-based approach must allow optimizations for processing spatial alarms installed by multiple mobile clients, whereas a client-based approach focuses more on energy-efficient solutions for evaluating a set of spatial alarms installed on a single client. Bearing in mind the problems inherent with the continuous spatial query evaluation approach and drawbacks of the periodic alarm evaluation approach, we develop a safe period-based alarm evaluation approach. The goal of applying safe period optimization is to minimize the amount of unnecessary alarm evaluations while ensuring zero or very low alarm miss rate. The other technical challenge behind safe period optimization is to minimize the amount of safe period computation, further improving system scalability and achieving higher throughput. We describe our basic approach for safe period computation in the next section and address the challenge of reducing the amount of safe period computations in Section 3. We evaluate the scalability and accuracy of our approach using a road network simulator and show that our proposed framework offers significant performance enhancements for the alarm processing server while maintaining high accuracy of spatial alarms.

2 Safe Period Computation

Safe period is defined as the duration of time for which the probability of an alarm being triggered for a subscriber is zero. Consider a subscriber S_i $(1 \leq i \leq N)$ and a spatial alarm A_j $(1 \leq j \leq M)$, where N is the total number of mobile users and M is the total number of alarms installed in the system. The safe period of alarm A_j with respect to subscriber S_i, denoted by $sp(S_i, A_j)$ can be computed based on the distance between the current position of S_i and the alarm region R_j, taking into account the motion characteristics of S_i and alarm target of A_j. Concretely, for alarms with mobile subscribers and static targets, the two factors that influence the computation of safe period $sp(S_i, A_j)$ are (i) the velocity-based motion characteristic of the subscriber S_i, denoted by $f(V_{S_i})$ and (ii) the distance from the current position of subscriber S_i to the spatial region R_j of alarm A_j, denoted by $d(S_i, R_j)$. Thus the safe period $sp(S_i, A_j)$ can be computed as follows:

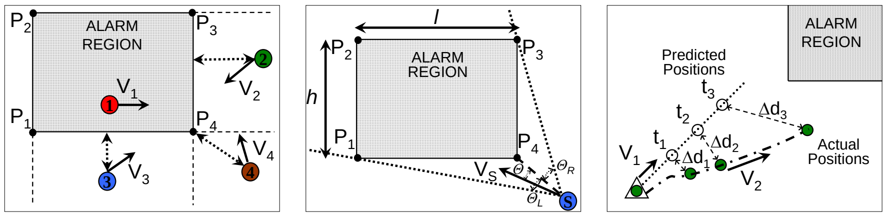
$$sp(S_i, A_j) = \frac{d(S_i, R_j)}{f(V_{S_i})} \tag{1}$$

2.1 Distance Measurements

We use *Euclidean distance* as the basic distance measure for safe period computation. It measures the minimum distance from the current position of the mobile user, denoted as $P_m = (x_m, y_m)$, to the spatial alarm region R. Consider a spatial alarm region R covering the rectangular region represented by four vertices of a rectangle (P_1, P_2, P_3, P_4), as shown in Figure 1(a). The minimum Euclidean distance from P_m to the spatial alarm region R, denoted by $d_{m,R}$, can be computed by considering the following four scenarios: ① when the mobile subscriber lies inside the spatial alarm region the distance $d_{m,R}$ is zero; ② when the mobile subscriber is within the y scope of the spatial alarm region, the minimum euclidean distance is the distance from the mobile subscriber to the nearer of the two spatial alarm edges parallel to the x-axis; ③ when the mobile subscriber is within the x scope of the spatial alarm region, minimum euclidean distance is the distance from the mobile subscriber to the nearer of the two spatial alarm edges parallel to the y-axis; and ④ when the mobile subscriber is outside both the x and y scope then the distance is the minimum of the euclidean distance to the four vertexes. Formally, $d_{m,R}$, the minimum Euclidean distance from mobile position P_m to the spatial alarm region R, is computed using the following formula:

$$d_{m,R} = \begin{cases} 0, & x_1 \leq x_m \leq x_2 \\ & and\ \ y_1 \leq y_m \leq y_2 \\ min(|x_m - x_1|, |x_m - x_2|), & y_1 \leq y_m \leq y_2\ only \\ min(|y_m - y_1|, |y_m - y_2|), & x_1 \leq x_m \leq x_2\ only \\ min(d_{m,1}, d_{m,2}, d_{m,3}, d_{m,4}), & otherwise \end{cases}$$

$d_{m,k}$, $k \in \{1, 2, 3, 4\}$ denotes the Euclidean distance from P_m to rectangle vertex P_k. The distance function $d_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ is used to compute the Euclidean distance between two points P_i and P_j.



(a) Euclidean Distance (b) Steady Motion (c) Dead Reckoning

Fig. 1. Basic Safe Period Computation

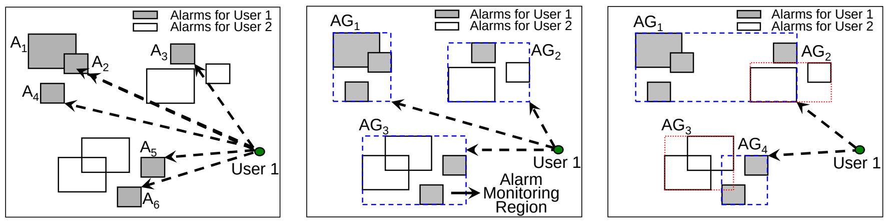
The safe period formula in equation 1 assumes that the subscriber heads towards the alarm region in a straight line along the direction of the minimum Euclidean distance, an assumption that rarely holds true. One way to relax this stringent condition is to use the *steady motion assumption*: If the subscriber is heading towards the alarm region R, then the deviation in her motion direction is not likely to be extreme. Figure 1(b) shows a scenario where the bounded deviation in subscriber motion is taken into account for calculating average safe period for subscriber S approaching alarm region R. In order for the subscriber S to enter the alarm region R at some future time instant, the average angle of motion for the subscriber S over the safe period must lie between $-\theta_L$ and $+\theta_R$ (as shown in the figure), which we refer to as *alarm trigger angular range*. Assume that the mobile subscriber heads towards the alarm region R in a direction at an angle θ to the minimum Euclidean distance vector; we refer to the distance from the subscriber position to the alarm region as the steady motion distance, denoted as $sm_{dist}(\theta)$. The steady motion-based safe period can be determined by $sm_{dist}(\theta)/f(V_S)$. Using the average steady motion distance obtained by computing $sm_{dist}(\theta)$ over all θ values ranging from $-\theta_L$ to $+\theta_R$, the steady motion-based safe period over the alarm trigger angular range can be calculated as,

$$sp = \frac{\int_{-\theta_L}^{+\theta_R} sm_{dist}(\theta) d\theta}{f(V_S) \int_{-\theta_L}^{+\theta_R} d\theta} = \frac{l + h}{f(V_S)(\theta_R + \theta_L)}, \tag{2}$$

where l, h denote the length and height of the spatial alarm region. The steady motion assumption provides a more realistic and optimistic measure for safe period computations compared to the minimum Euclidean distance approach.

2.2 Velocity Measurements

The use of maximum travel speed of the mobile client for the velocity function $f(V_S)$ carries both advantages and disadvantages. On one hand, the 'maximum travel speed' can be set by pre-configuration based on a number of factors, such as the nature of the mobile client (car on the move or a pedestrian walking on the street), or the type of road used. On the other hand, maximum speed-based estimation is often pessimistic, especially in the following two scenarios: (i) when the mobile client stops for an extended period of time, or (ii) when the mobile client suddenly turns onto a road with very low speed limit. Another



(a) Per Alarm Safe Period (b) SL Grouping (c) SSSL Grouping

Fig. 2. Alarm Locality-based Grouping

issue related to the use of maximum speed of a mobile client for the velocity function $f(V_S)$ is related to alarm misses. The maximum velocity-based approach may fail to trigger alarms in cases where the maximum speed for the mobile subscriber increases suddenly. For example, a vehicle moving from a street onto a state highway would experience a sudden increase in its velocity, which may invalidate safe period computations. One way to address such sudden increase in velocity is to use *dead reckoning* techniques which require the mobile user to report to the server when her velocity increases over a certain threshold, as shown in Figure 1(c). The use of dead reckoning or similar techniques will allow the server to recompute the safe period for mobile client upon any significant velocity change. In Figure 1(c), the mobile client keeps track of its predicted positions based on its maximum speed and its actual positions. As soon as the difference between the predicted position and the actual position exceeds a given threshold value (say δ), the client provides its current speed to the server.

2.3 Safe Period-based Alarm Evaluation

The safe period-based approach processes a spatial alarm in three stages. First, upon the installation of a spatial alarm, the safe period of the alarm with respect to each authorized subscriber is calculated. Second, for each alarm-subscriber pair, the alarm is processed upon the expiration of the associated safe period and a new safe period is computed. In the third stage, a decision is made regarding whether the alarm should be fired or wait for the new safe period to expire.

When compared to periodic alarm evaluation, the safe period approach for spatial alarm processing reduces the amount of unnecessary alarm evaluation steps, especially when the subscriber is far away from all her alarms. On the other hand, the main cost of the basic safe period approach described in this section is due to the excessive amount of unnecessary safe period computations, as the basic safe period approach performs safe period computation for each alarm-subscriber pair. One obvious idea to reduce the amount of unnecessary safe period computations is to group spatial alarms based on geographical proximity and calculate safe period for each subscriber and alarm group pair instead of each alarm-subscriber pair.

3 Alarm Grouping Techniques

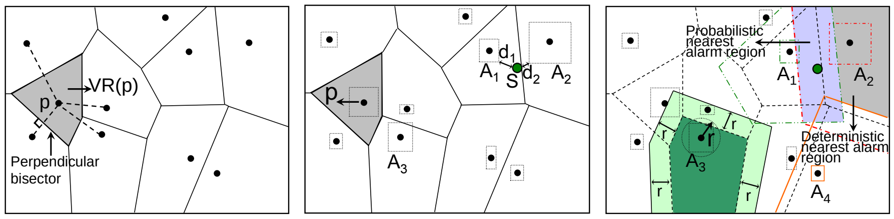
The basic premise behind alarm grouping is to reduce the number of safe period computations while ensuring no alarm misses. In this section, we present three alternative grouping techniques, each of which offers different degree of improvement for safe period computations. First, we group all alarms based on their spatial locality. Alternatively, we apply spatial locality based-grouping to alarms of each individual subscriber. Our experimental study shows that this approach is more effective. The third locality-based alternative is to employ the nearest alarms-based grouping, which is effective but costly when there are frequent alarm additions and removals.

3.1 Spatial Locality-based Grouping

Spatial locality-based (SL) grouping considers the set of alarms from all users and groups together the nearby alarms. This approach outperforms basic safe period alarm evaluation if each group has a large number of alarms belonging to the same subscriber. Figure 2(a) displays the alarm regions for a set of installed alarms. The alarms for user 1 are marked by shaded regions. Basic safe period evaluation computes the distance from each of the six alarms $\{A_i \mid 1 \leq i \leq 6\}$. In comparison, Figure 2(b) shows three groups derived from spatial locality-based grouping technique. We use a simple R-tree implementation in order to group alarms and identify the *minimum bounding rectangles (MBRs)* for alarm groups which are also referred to as *alarm monitoring regions*. Instead of computing distance for each alarm-subscriber pair, spatial locality-based grouping calculates the distance for each subscriber and alarm group pair. However, on entering a monitoring region the distance to all relevant alarms within the alarm group also needs to be computed. Despite this additional evaluation step, the number of safe period computations may be considerably reduced by grouping alarms according to spatial locality. Instead of six computations required by the basic safe period technique, only three computations need to be performed as all three alarm groups, $\{AG_i \mid 1 \leq i \leq 3\}$, contain alarms relevant to user 1. Further computations are dependent on the number of relevant alarms within the users' current alarm monitoring region. Even though this approach reduces the number of computations it requires considerable additional processing to determine the set of relevant alarm groups for each subscriber and the set of relevant alarms for each subscriber within an alarm group. The lack of subscriber-specificity in the underlying data structure, R-Tree, leads to retrieval of large number of unnecessary alarms. This technique proves to be efficient in presence of large number of public alarms as the effect of subscriber-specificity is reduced in this situation.

3.2 Subscriber-Specific Spatial Locality-based Grouping

In contrast to spatial locality-based grouping, subscriber-specific spatial locality-based (SSSL) grouping performs a two level grouping: the first level grouping is on all subscribers and the second level grouping is on spatial alarms relevant to each subscriber. We use a B-tree based implementation to speed up search on subscribers and an R-Tree implementation to capture spatial locality of alarms for each subscriber. The underlying data structure is a hybrid structure which uses a B-tree for subscriber search at the first level and an R-tree for subscriber



(a) Voronoi Diagram (b) Voronoi Extension (c) Voronoi Extension

Fig. 3. Nearest Alarms-based Grouping

specific spatial alarm search at the second level. Figure 2(c) shows an example of this grouping. Alarms installed by user 1 are grouped together in AG_1 and AG_4 and may be fired only when the user is entering the $MBRs$ of AG_1 or AG_4. Subscriber specific spatial locality-based grouping has two advantages over the previous approaches. First, the number of safe period computations is significantly reduced. Second, each alarm group contains alarms relevant to a single user, thus no irrelevant processing is performed. Our experimental results show that this approach is efficient in the presence of large number of subscribers and for large number of private and shared alarms.

3.3 Nearest Alarms-based Grouping

Nearest alarms-based grouping allows the system to perform one or only a few alarm checks dependent on the current subscriber position. The idea is to have each location on the map associated with the nearest spatial alarm region(s). In order to perform nearest alarms-based grouping we use an extension of the well known Voronoi diagram geometric structure [5]. The Voronoi diagram for a given set of points P in d-dimensional space $\mathbb{R}^d$ partitions the space into regions where each region includes all points with a common closest point ϵ P. The *Voronoi region VR(p)* corresponding to any point $p \; \epsilon \; P$ contains all points $p_i \; \epsilon \; \mathbb{R}^d$ such that,

$$\forall p' \epsilon P, p' \neq p, dist(p_i, p) \leq dist(p_i, p') \tag{3}$$

Figure 3(a) shows the Voronoi diagram for a set of points in two-dimensional space $\mathbb{R}^2$ with euclidean distance metric. The shaded area marks out the *Voronoi region VR(p)* for the point p.

In order to create a Voronoi diagram for spatial alarms we first represent each spatial alarm region R by its center point (x_{cr}, y_{cr}) and l, h representing the length and height of the alarm region. We consider the center point of each alarm region as a Voronoi site and create the Voronoi diagram as shown in Figure 3(b). But alarm regions may overlap with adjacent Voronoi regions as for alarm A_3 in the figure. Also, consider the subscriber S in the figure residing in the Voronoi region of alarm A_1. S is at a minimum Euclidean distance d_1 from the alarm region of A_1 and at a minimum Euclidean distance d_2 to the alarm region of A_2. Even though $d_2 < d_1$, A_1 is incorrectly identified as the nearest alarm on the basis of the underlying Voronoi diagram. In order to rectify this problem, we introduce an extension to the original Voronoi diagram by extending the boundary of each Voronoi region by the *extension radius* r associated with each point p where $r =$

$\sqrt{(\frac{l}{2})^2 + (\frac{h}{2})^2}$. l, h denote the length and height of the alarm region associated with center point p. The extended Voronoi regions for alarms A_1, A_2, A_3 and A_4 are shown in Figure 3(c). Extending the Voronoi region boundaries leads to overlaps among neighboring Voronoi regions; subscribers inside overlapping regions (*probabilistic nearest alarm region*) may have more than one possible nearest alarm whereas subscribers inside non-overlapping regions (*deterministic nearest alarm region*) can have only one nearest alarm.

Nearest alarm grouping is efficient for systems that have infrequent addition or removal of alarms and have no hotspots. However, it fails when there is frequent addition/removal of spatial alarms, since Voronoi diagrams need to be reconstructed each time an alarm is added or removed. In addition, high density of alarms in some areas may also lead to large overlaps among Voronoi regions, reducing the efficiency of this technique.

4 Experimental Evaluation

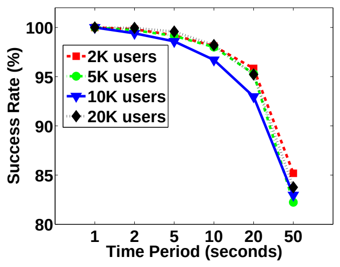
In this section, we report our experimental evaluation results. We show that our safe period-based framework and optimization techniques for spatial alarm processing are scalable while maintaining high accuracy.

4.1 Experimental Setup

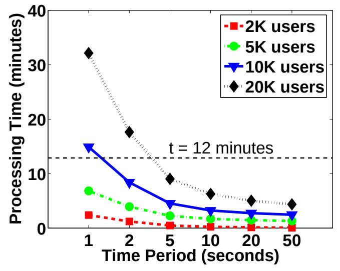
Our simulator generates a trace of vehicles moving on a real-world road network using maps available from the National Mapping Division of the U.S. Geological Survey (USGS [4]) in Spatial Data Transfer Format (SDTS [3]). Vehicles are randomly placed on the road network according to traffic densities determined from the traffic volume data in [9]. We use a map from Atlanta and surrounding regions of Georgia, which covers an area larger than 1000 km^2, to generate the trace. Our experiments use traces generated by simulating vehicle movement for a period of fifteen minutes, results are averaged over a number of such traces. Default traffic volume values allow us to simulate the movement of a set of 20,000 vehicles. The default spatial alarm information consists of a set of 10,000 alarms installed uniformly over the entire map region; around 65% of the alarms are private, 33% shared and the rest are public alarms.

4.2 Experimental Results

The first set of experiments measures the performance of periodic alarm evaluation by varying the time period of updates and shows that this approach does not scale. The second set of experiments compares the basic safe period approach against periodic evaluation and shows that safe period-based alarm processing offers higher success rate with lower evaluation time. The last set of experiments compares the performance of the various grouping-based optimizations against the basic safe period approach exhibiting the scalability of our grouping optimizations.

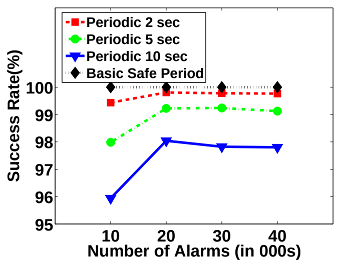


(a) Success Rate

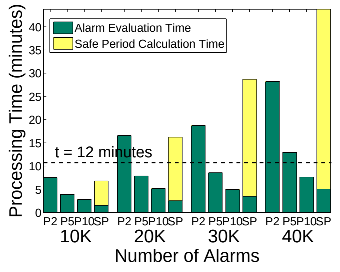


(b) Alarm Processing Time

Fig. 4. Scalability with Varying Number of Users



(a) Success Rate

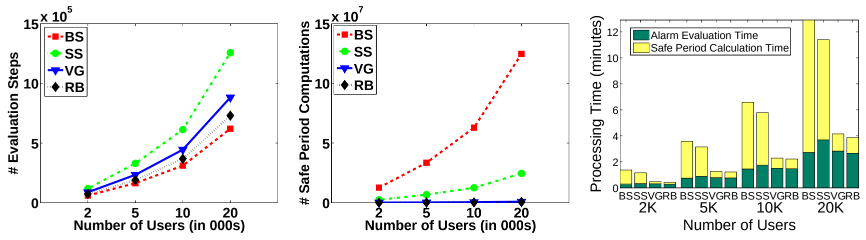


(b) Alarm Processing Time

Fig. 5. Safe Period Optimization with Varying Number of Alarms

Scalability Problems of Periodic Alarm Evaluation Technique: In this first set of experiments, we measure the scalability of the periodic alarm evaluation technique with varying number of users. Figure 4 displays the results as we vary the number of users from 2K to 20K. The time period t_p for periodic alarm evaluation is varied from 1 second to 50 seconds. As can be seen from Figure 4(a), the success rate for alarm evaluation is 100% only if t_p= 1 second; for higher t_p success rate starts falling, even with t_p= 2 seconds the success rate does fall to 99.9% which may not be acceptable from QoS viewpoint as this translates to a significant number of alarm misses. The sequence of alarms to be triggered for 100% success rate are determined from a trace generated with highly frequent location updates for each user. The alarm processing time is plotted in Figure 4(b). Our traces are of fifteen minutes duration; considering that the system has around 80% of this time for processing spatial alarms we set the maximum processing time available to the system at t=12 minutes as indicated by the horizontal dotted line in Figure 4(b). For 10K users the system is unable to process alarms at t_p=1 seconds, thus failing to attain 100% success rate. For 20K users, this scalability problem becomes worse and the system is able to evaluate alarms only at t_p=5 seconds. Thus, we conclude that periodic evaluation approach does not scale.

Performance Comparison with Basic Safe Period Approach: In this section, we compare the performance of basic safe period approach against periodic evaluation. We display the results for periodic approach with t_p=2 seconds, t_p= 5 seconds, t_p=10 seconds and the basic safe period optimization as discussed in Section 2 (P2, P5, P10 and SP in Figure 5(b)). Figure 5 displays the success rate



(a) Alarm Evaluation Steps (b) Safe Period Computations (c) Alarm Processing Time

Fig. 6. Safe Period Optimizations with Varying Number of Users

and processing time as we increase the number of alarms from 10K to 40K. Figure 5(a) displays that the success rate is 100% for basic safe period approach and all periodic approaches miss at least a few alarm triggers. Figure 5(b) displays the alarm processing time for P2, P5, P10 and SP with varying number of alarms. The alarm processing time, as shown in Figure 5(b), displays the inability of our basic safe period approach to scale to large number of alarms. In presence of even 20K installed alarms, the approach has excessive safe period computation time which pushes the overall processing time beyond the 12 minute limit determined earlier. Our alarm grouping and subscriber mobility-based techniques provide optimizations to overcome this problem.

Scalability of Safe Period Evaluation Techniques: We now discuss the performance of the safe period optimization techniques to test the scalability of our framework. Figure 6 shows the number of alarm evaluation steps, number of safe period computations and the alarm processing time required by each approach: Basic Safe Period Optimization (BS), Subscriber-Specific Spatial Locality (SS), Voronoi Grid-Based (VG) and a Range-based Subscriber-Specific Grouping Optimization (RB). VG and RB approaches consider alarms only in the vicinity of the current subscriber position for safe period computation. Results for Spatial Locality-based grouping show expected trends but this approach has high overall processing time as the system needs to perform significant amount of computation to determine relevance of alarms/alarm groups for each subscriber. Hence, we exclude this approach from the results.

Figure 6(a) displays the number of alarm evaluation steps required by each approach. Basic safe period measures the safe period to each relevant alarm and uses this safe period to avoid further evaluations. As a result, this approach has to perform a low number of alarm evaluations but each evaluation step involves a very large number of safe period computations. Hence the number of safe period computations for this approach is extremely large (Figure 6(b)) which makes this approach overall computationally expensive as can be seen from the total alarm processing times in Figure 6(c). Subscriber-specific spatial locality grouping incurs a large number of alarm evaluation steps as can be seen from Figure 6(a). This approach first evaluates safe period for each alarm group; once the user enters an alarm monitoring region another evaluation step is required to determine the safe period to all alarms lying within the alarm monitoring region. Further, this approach needs to keep a check on subscriber position inside the alarm monitoring region and switch to per alarm group-based safe period computations once the subscriber moves outside the current alarm monitoring region. These additional evaluation steps imply that this approach will incur a larger number of alarm evaluation steps with each evaluation step requiring a small number of safe period computations: either for each alarm group or for all alarms lying within the current alarm monitoring region. Thus the number of safe period computations required by this approach is much lower than the basic approach despite the larger number of alarm evaluation steps. Consequently, the overall processing time for SS is lower than the BS approach as can be seen from Figure 6(c). The VG and RB approaches lower the number of alarm evaluation steps by considering only alarms or alarm groups in the vicinity of the client. In this set of experiments, the RB approach considers alarms within a radius of 1000m from the client position. VG approach overlays a grid with cell size 1000m $\times$ 1000m on top of the Voronoi extension and considers alarms only within the current subscriber grid cell. The number of evaluation steps for these approaches is still larger than the number of evaluation steps used by the basic approach as the safe periods computed by this approach may be lower than the safe period computed by the basic approach, in case no relevant alarms/alarm groups lie within the radius range or the current grid cell of the subscriber. However, each alarm evaluation step involves a very small number of safe period calculations leading to an extremely small number of safe period computations (in Figure 6(b) results for VG and RB are overlapping and values are much smaller than other two approaches). Consequently, the overall processing times for these two approaches are significantly lower than other approaches. From these results we can conclude that our safe period optimizations significantly aid the scalability of the system.

5 Related Work

An event-based location reminder system has been advocated by many human computer interaction projects [12, 14, 8, 13, 10]. Understandably, the primary focus of the work is from the point of view of the usability of such systems. None of these approaches deal with the system oriented issues which need to be resolved to make such systems feasible. In the realm of information monitoring, event-based systems have been developed to deliver relevant information to users on demand [11, 7]. In addition to monitoring continuously evolving user information needs, spatial alarm processing systems also have to deal with the complexity of monitoring user location data in order to trigger relevant alerts in a non-intrusive manner. Applications like Geominder [1] and Naggie [2] already exist which provide useful location reminder services using cell tower ID and GPS technology, respectively. Client-based solutions for spatial alarm processing should focus on efficiently evaluating spatial alarms while preserving client energy. Our server-centric architecture makes it possible for users to share alarms and make use of external location information monitoring services which provide relevant location-based alerts. A server-centric approach is also essential for extending the technology to clients using cheap location detection devices which may not possess significant computational power.

6 Conclusion

The paper makes two important contributions towards supporting spatial alarm-based mobile applications. First, we introduce the concept of *safe period* to minimize the number of unnecessary alarm evaluations, increasing the throughput and scalability of the system. Second, we develop a suite of spatial alarm grouping techniques based on spatial locality of the alarms and motion behavior of the mobile users, which reduces the safe period computation cost for spatial alarm evaluation at the server side. We evaluate the scalability and accuracy of our approach using a road network simulator and show that the proposed safe period-based approach to spatial alarm processing offers significant performance enhancements for alarm processing on server side while maintaining high accuracy of spatial alarms.

Acknowledgements

This work is partially supported by grants from NSF CyberTrust, NSF SGER, NSF CSR, AFOSR, IBM SUR grant and IBM faculty award. The authors would like to thank Bugra Gedik for providing the mobile object simulator.

References

1. Geominder. `http://ludimate.com/products/geominder/`.
2. Naggie 2.0: Revolutionize Reminders with Location! `http://www.naggie.com/`.
3. Spatial Data Transfer Format. `http://www.mcmcweb.er.usgs.gov/sdts/`.
4. U.S. Geological Survey. `http://www.usgs.gov`.
5. F. Aurenhammer. Voronoi Diagrams – A Survey of a Fundamental Geometric Data Structure. *ACM Computing Surveys*, 23(3):345–405, 1991.
6. B. Bamba, L. Liu and P.S. Yu. Scalable Processing of Spatial Alarms. *Technical Report, Georgia Institute of Technology*, 2008.
7. V. Bazinette, N. Cohen, M. Ebling, G. Hunt, H. Lei, A. Purakayastha, G. Stewart, L. Wong, and D. Yeh. An Intelligent Notification System. *IBM Research Report RC 22089 (99042)*, 2001.
8. A. Dey and G. Abowd. CybreMinder: A Context-Aware System for Supporting Reminders. In *Second International Symposium on Handheld and Ubiquitous Computing*, pages 172–186, 2000.
9. M. Gruteser and D. Grunwald. Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking. In *MobiSys*, 2003.
10. S. Kim, M. Kim, S. Park, Y. Jin, and W. Choi. Gate Reminder: A Design Case of a Smart Reminder. In *Conference on Designing Interactive Systems*, pages 81–90, 2004.
11. L. Liu, C. Pu, and W. Tang. WebCQ - Detecting and Delivering Information Changes on the Web. In *CIKM*, pages 512–519, 2000.
12. P. Ludford, D. Frankowski, K. Reily, K. Wilms, and L. Terveen. Because I Carry My Cell Phone Anyway: Functional Location-Based Reminder Applications. In *SIGCHI Conference on Human Factors in Computing Systems*, pages 889–898, 2006.
13. N. Marmasse and C. Schmandt. Location-Aware Information Delivery with ComMotion. In *HUC*, pages 157–171, 2000.
14. T. Sohn, K. Li, G. Lee, I. Smith, J. Scott, and W. Griswold. Place-Its: A Study of Location-Based Reminders on Mobile Phones. In *UbiComp*, 2005.