Cost-Effective Multidimensional Publish/Subscribe Services in Sensor Networks

Duc A. Tran   Cuong H. Pham
Department of Computer Science
University of Massachusetts, Boston, MA 02125
Email: {duc, cpham}@cs.umb.edu

***Abstract*—Publish/subscribe services are natural applications of sensor networks as sensors are designed mainly to detect and notify upon events of interest. To enable these services, the brokerage mechanism that routes a given event to the subscribing nodes must be confined to the network's resource constraints. GHT has been used as a cost-effective way to disseminate subscriptions and events to the sensor network. It is, however, efficient only for exact-match subscriptions. We propose an efficient GHT-like technique for both exact-match and range subscriptions where the data space can be of any dimension.**

I. Introduction

With the emergence of sensor technologies, which allows for cheaper sensors with increasing sensing capability, there is a growing trend of deploying sensor nodes in the physical world to form a network for monitoring the environment. Important to such a sensor network is a publish/subscribe functionality that enables the users to subscribe to events of interest and the sensor nodes to publish their events, such that the users can be notified upon occurrences of matching events. In this context, the users and sensors are called the subscribers and publishers, respectively. For example, in the application of disaster monitoring, a subscriber can submit to the network in advance a query specifying the early warnings of a wildfire, so that as soon as an event matching these warnings is detected by a sensor, the event is propagated to the network to notify the subscriber. In another example, we can help our soldiers navigate safely on a battlefield by deploying a sensor network that is notified upon events implying the existence of enemy forces.

Despite various efforts, research on publish/subscribe mechanisms for sensor networks remains infancy. It is understandable, though. Firstly, unlike publish/subscribe systems on the Internet (e.g., [1]–[3]), that designed for sensor networks faces unseen challenges due to limitations in sensor storage, processing, and communication capacities. Secondly, the publish/subscribe model is much different from the traditional search model which has widely been addressed for sensor networks [4]–[10]. The traditional search follows the request/response model, where a search query is submitted on demand expecting the matching results to return immediately. In contrast, a query in the publish/subscribe model is submitted and stored a priori, for which the matching results may or may not already exist; in the latter case, the query subscriber will be notified when the matching results will become available. Thus, a main problem for the traditional search systems is to manage the existing data in advance for fast data retrieval at a later time, while the corresponding problem for publish/subscribe systems is to manage the submitted queries in advance for fast query matching in the future.

For any publish/subscribe service in a sensor network, it is desirable that the following costs are low: the *replication cost* to replicate queries in the network and the *computation cost* to check query/event matching conditions. In addition, it should take short *notification delay*; i.e., short time between when an event is published until the publication reaches its subscribing nodes.

However, the time requirement and the cost requirement are conflicting with each other. Indeed, to increase the chance for an event to meet its matching queries quickly, the query should be replicated at an sufficiently large number of nodes, which, however, results in large communication, storage, and computation costs. As will be discussed in the next section, existing techniques are either time-sensitive [10]–[12] or cost-effective [5], or a compromise in between [10].

In this paper, we focus on the costs. The main approach to lowering these costs is to use a GHT method [5] to map a query and an event to the same geographic coordinate of the sensor network if they match. Although GHT is well-known, its application for publish/subscribe services in sensor network has been proposed only for the case where queries are point queries not range queries [5], [10]. For example, while queries like *"notify me of the sensor locations when their temperature is 95^oF"* are addressed, those queries like *"notify me of the sensor locations when their temperature is between 90^oF and 100^oF"* are not. This is because when GHT is used to hash a range query, the corresponding geographic area can be large; hence, expensive replication and computation costs.

We propose a GHT-like solution that works efficiently for multidimensional range queries. Our hashing method is based on random projections to the sensor location space. To reduce the replication cost, many publish/subscribe techniques take advantage of subscription coverings to remove unnecessary replications [13], [14]. However, not many coverings can be found in high dimension. We instead use the covering relationship among their projections, rather than the original-subscription covering relationships. To reduce the computation cost, the queries are stored in the network such that, given an event, the computation checking the matching conditions

1-4244-2575-4/08/$20.00 ©2008 IEEE

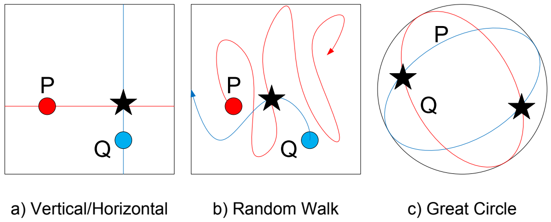


Fig. 1. Double-Ruling: Query path and event path (almost) certainly meet

takes places only at nodes highly likely of storing the matching queries.

The remainder of the paper is organized as follows. We discuss the related work in Section 2. We present the details of our technique in Section 3. The evaluation results are reported in Section 4. We conclude the paper in Section 5.

II. Related Work

Subscribers and publishers do not know each other. The simplest way for them to find each other is via broadcast. In this approach, a subscription is sent to all sensors in a broadcast tree [15], or, alternatively, each event is sent to all possible subscriber nodes [16]. Either way, the communication cost due to broadcasting can be prohibitively high for large networks.

The double-ruling approach [10]–[12] can be used to avoid broadcasting. The basic idea is to replicate each subscription on one (muti-) path, each event on another, such that an event path and a subscription path always or highly likely meet. For example, one can use Random Walks [17] or Rumor Routing [11] to replicate a subscription or to route an event (Figure 1(b)). Since it is highly probable that two random walks intersect, an event will almost certainly meet its matching subscriptions.

Another double-ruling technique is to use horizontal and vertical lines for query and event dissemination, respectively (Figure 1(a)). Since any vertical line must meet every horizontal line, an event will find all the subscriptions matching it. This technique is time-sensitive because it guarantees that the number of hops the event visits on its way from the publisher to the subscriber is their Manhattan distance (which is less than a small multiple of the corresponding Euclidean distance).

The notification time can be further improved as in the double-ruling technique proposed in [12]. This technique maps the sensor nodes virtually onto the surface of a 3-D sphere and uses this sphere's great circles to advertise subscriptions and events because any two great circles always meet (Figure 1(c)).

The double-ruling approach is not cost-effective. For example, using 2D horizontal/vertical lines, each subscription is replicated at approximately $\sqrt{n}$ nodes for a network of n sensors uniformly distributed. The corresponding cost is $\sqrt{n\pi}$ using 3D great circles as in [12] and higher for random walks. The computation cost is also as high because the matching conditions are checked at every node visited by any given event.

Geographic Hash Table (GHT) [5] is an approach that hashes a single value (called "key") to a geographic coordinate for efficient storage and retrieval. Thus, if both events and subscriptions each can be represented as a single key, we can use GHT. A subscription with key k will be stored at the sensor closest to location $h(k)$ according to the hashing on k. When an event with the same key k emerges, it will be routed to the node closest to location $h(k)$ where it can find all the matching subscriptions.

Using GHT, both the replication cost and computation cost are modest because a subscription query is replicated at only one node and the matching computation also takes place at one node. A combination of hashing with double-ruling is proposed in [10]. Although this is a nice technique aimed at reducing both time and costs, it does not address the case where a subscription query corresponds to a range of values (so neither GHT). Moreover, this technique assumes a number of known landmarks, which we do not require with our technique.

III. The Proposed Approach

Without loss of generality, suppose that the universe of events is the d-dimension unit cube $\mathcal{D} = [0,1]^d$, where d is the number of attributes associated with each event. For example, if the sensor network is used to monitor temperature, humidity, wind speed, and air pressure of some area, d is four – representing those four sensor data attributes.

A subscription can be any range of points in d dimensions. However, for easy implementation, it is usual that a subscription is represented by a hyperrectangle or a hypersphere (rectangle or sphere, in short). For example, a rectangular subscription can be *"find all sensor locations with temperature above 100^oF, humidity below 20%, wind speed above 50 mph and air pressure below 1 atm"*. It is sometimes a tedious process to specify all the lower and upper bounds for all the attributes of a query. In such cases, it is more convenient to provide an event sample and request all the events similar to this sample. A query of this kind can be realized by a spherical subscription in which the sample is the center of the sphere and similarity is constrained by the sphere's radius. In this paper, we assume the spherical subscription form and represent each subscription $q = (s, r)$ by a sphere centered at $s \in \mathcal{D}$ with radius $r > 0$.

We also assume a sensor network where the location of each sensor node V is represented as a 2D point $Point(V) \in \mathcal{C} = [0,1]^2$ (our technique can be generalized to also work for sensors whose locations are 3D. Routing in the sensor network from one location to another can be done by a geometrical routing protocol [18]. Our idea is based on the use of random projection to map the subscription/event space into the 2D sensor space.

A. Preliminary

Let $\{\overrightarrow{u_1}, \overrightarrow{u_2}\}$ be two random d-dimension orthonormal vectors. Consider a subscription query $q = (s, r)$, which is a sphere centered at point $s \in \mathcal{D}$ with radius $r \geq 0$. Projecting

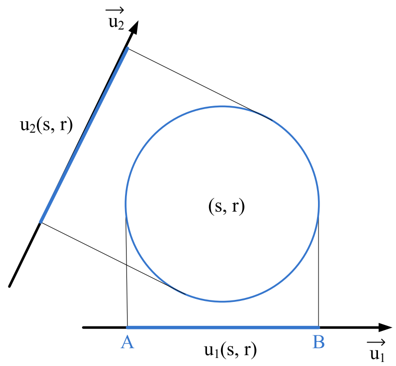


Fig. 2. Projection of a spherical subscription on two random dimensions

this sphere on to the two random vectors, we obtain the following rectangle: (see Figure 2)

$$(s,r) \rightarrow u(s,r) = \prod_{i=1}^{2} [\langle u_i, s\rangle - r, \langle u_i, s\rangle + r]$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. To make this rectangle fit in $[0,1]^2$, we apply the following linear transformation. Suppose that r_{max} is the maximum subscription radius possible. Letting $\alpha_i = -r_{max} + \sum_{j=1}^{d} \min(0, u_{ij})$ and $\beta_i = r_{max} + \sum_{j=1}^{d} \max(0, u_{ij})$ the following rectangle

$$u_{\mathcal{C}}(q) = \prod_{i=1}^{2} \left[\frac{\langle u_i, s\rangle - r - \alpha_i}{\beta_i - \alpha_i}, \frac{\langle u_i, s\rangle + r - \alpha_i}{\beta_i - \alpha_i} \right]$$

which is linearly transformed from $u(s,r)$, is inside $[0,1]^2$. We refer to this rectangle as the *projection* of query q. The center of this projection is the 2D point

$$Center_{\mathcal{C}}(q) = \left(\frac{\langle u_1, s\rangle - \alpha_1}{\beta_1 - \alpha_1}, \frac{\langle u_2, s\rangle - \alpha_2}{\beta_2 - \alpha_2} \right)$$

Similarly, using the same projection method, each event $x \in \mathcal{D}$ is mapped to the following point in the sensor field (imagine x as a zero-radius query $(x, 0)$):

$$x \rightarrow u_{\mathcal{C}}(x) = \left(\frac{\langle u_1, x\rangle - \alpha_1}{\beta_1 - \alpha_1}, \frac{\langle u_2, x\rangle - \alpha_1}{\beta_1 - \alpha_1} \right)$$

The following properties are observed:

- Property (1): If an event x satisfies a query q, then $u_{\mathcal{C}}(x) \in u_{\mathcal{C}}(q)$
- Property (2): Given two events x and y, their similarity is preserved under the projection. Indeed, denoting the Euclidean distance by $d(.,.)$, we have $d(u_{\mathcal{C}}(x), u_{\mathcal{C}}(y)) = \sqrt{\sum_{i=1}^{2} \left(\frac{\langle u_i, x-y\rangle}{\beta_i - \alpha_i} \right)^2} \leq \sqrt{\sum_{i=1}^{2} \left(\frac{\|x-y\|}{2r_{max} + \sum_{j=1}^{d} |u_{ij}|} \right)^2} = d(x,y) \sqrt{\sum_{i=1}^{2} \frac{1}{(2r_{max} + \sum_{j=1}^{d} |u_{ij}|)^2}}$
- Property (3): Given two overlapping queries q and q', we have $u_{\mathcal{C}}(q \cap q') \subseteq u_{\mathcal{C}}(q) \cap u_{\mathcal{C}}(q')$. Therefore, if we define the similarity between two subscription queries by the number of common events, it is preserved under the projection.

B. Query Subscription

The three properties suggest that we store a query q in sensors inside the region $u_{\mathcal{C}}(q)$, and advertise an event x to the sensor closest to $u_{\mathcal{C}}(x)$. Using this strategy, because of Property (1), it is highly likely that query q will be notified of x if x satisfies q. In addition, Properties (2) and (3) imply that similar events and similar subscriptions are mapped to sensors that are nearby each other.

The geometric area of the region $u_{\mathcal{C}}(q)$ is

$$\frac{4r^2}{(2r_{max} + \sum_{j=1}^{d} |\, u_{1j} \,|)(2r_{max} + \sum_{j=1}^{d} |\, u_{2j} \,|)}$$

Thus, the replication cost is less if the queries are more specific (i.e., small r) or if the dimension d is higher. In the case of small d or large r, the region can be large. We propose to avoid unnecessary replications by taking advantage of the subscription covering relationship. Note that, if query q' covers query q, it must be true that $u_{\mathcal{C}}(q')$ covers $u_{\mathcal{C}}(q)$. In other words, if a new query is covered by an existing query, the nodes that the former query is mapped to must already store the existing query. Because those events that satisfy q' will be returned to notify q' anyway, which will be filtered to match q, there is no need to replicate query q further.

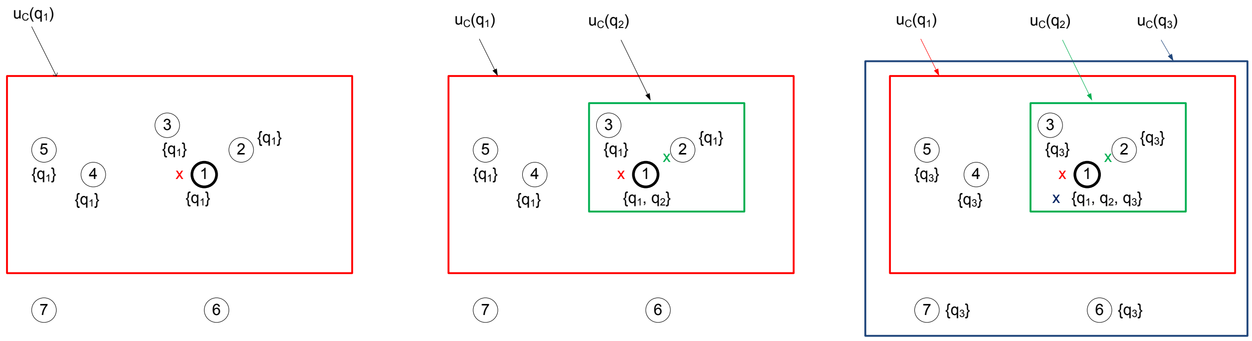
It should, however, be noted that as the data dimensionality increases, the number of subscription coverings is decreased because the subscription set is more sparse. As such, merely using this relationship might not be sufficiently effective in reducing the replication cost.

We propose that a new query q is not replicated if an existing query q' is found such that $u_{\mathcal{C}}(q') \supset u_{\mathcal{C}}(q)$ (instead of using the condition $q' \supset q$). In other words, we do not replicate a query if its projection is covered by the projection of an existing query. The likelihood of $u_{\mathcal{C}}(q') \supset u_{\mathcal{C}}(q)$ is much higher than that of $q' \supset q$, thus this strategy can reduce the replication cost significantly.

We associate with each query q with a node called the home node $home(q)$ – the node that stores the first copy of query q if it is replicated multiple times or the only copy otherwise. The protocol to replicate a query in the subscription strategy is as follows.

Protocol 3.1 (Subscription Protocol):

1) *Use the sensor routing protocol to send q to the node V_q such that $Point(V_q)$ is the closest to $Center_{\mathcal{C}}(q)$.*
2) *If there is a query q' currently stored at node V_q such that $u_{\mathcal{C}}(q') \supseteq u_{\mathcal{C}}(q)$, store query q at node $home(q')$*
3) *Else*
   a) *Set $home(q) = V_q$ and store q at V_q*
   b) *Send (q, V_q) to surrounding nodes V' such that $Point(V') \in u_{\mathcal{C}}(q)$. At each node V',*
      i) *For each existing query q' stored at V' such that $u_{\mathcal{C}}(q') \subseteq u_{\mathcal{C}}(q)$ and $home(q') = V_q$, remove query q' from node V'*



(a) Query q_1 is firstly submitted (b) After query q_2 is submitted (c) After query q_3 is submitted

Fig. 3. Subscription protocol example: The '×' mark represents the center of each subscription projection and associated with each node is the set of subscriptions it stores

An illustration is given in Figure 3. Suppose that queries q_1, q_2, q_3 are submitted into the network at times in that order. Query q_1 is submitted first (see Figure 3(a)), whose projection center $Center_{\mathcal{C}}(q_1)$ is closest to node 1 and also contains nodes 2, 3, 4, 5. Therefore, q_1 is stored at these nodes and the home node of q_1 is node 1. When q_2 is submitted (see Figure 3(b)), it is sent to node 2 because this node is the closest to the center of q_2. Because node 2 already stores query q_1 and $u_{\mathcal{C}}(q_1) \supseteq u_{\mathcal{C}}(q_2)$, query q_2 will be stored at the home node of query q_1, i.e., node 1 only; hence, a significant reduction in subscription load. When query q_3 is submitted (see Figure 3(c)), it is sent the closest node to its projection center, which is node 1. Because its projection is not covered by any other's, it is stored at node 1 and also at nodes 2, 3, 4, 5, 6, 7 that lie inside $u_{\mathcal{C}}(q_3)$. Node 1 serves as the home node of query q_3. At node 1, since $u_{\mathcal{C}}(q_3) \supseteq u_{\mathcal{C}}(q_1)$ and $u_{\mathcal{C}}(q_3) \supseteq u_{\mathcal{C}}(q_2)$, both queries q_1 and q_2 are removed from node 1. Query q_1 is also removed from nodes 2, 3, 4, and 5. Thus, we have avoided replications of these two queries.

C. Event Notification

When an event x occurs, the notification procedure is as follows.

Protocol 3.2 (Notification Protocol):

1) *Use the sensor routing protocol to advertise it to the node V_x such as $Point(V_x) \in u_{\mathcal{C}}(x)$*
2) *For each query q stored at node V_x such that $u_{\mathcal{C}}(x) \in u_{\mathcal{C}}(q)$, forward x to node $home(q)$ – the home node of q*
    a) *For each home node V that receives x, notify all queries that match x and that call V home*

We note that searching node V_x alone is insufficient to find all the matching queries of x. There may be subscriptions matching x, which are not stored at node V_x. These subscriptions are actually stored at the home nodes of the queries q at V_x such that $u_{\mathcal{C}}(x) \in u_{\mathcal{C}}(q)$. Thus, we have to visit the home nodes of such queries q to find all possible matching queries of x. For example, continuing the illustration earlier in Figure 3(c), suppose that an event x occurs that satisfies query q_2 and whose projection is closest to node 7. An advertisement will be sent to node 7. Node 7 finding that x satisfies query q_3 will forward the advertisement of x to the home node 1 of query q_3. At node 1, query q_2 will be found.

Given an event x, the computation to find queries matching it takes place only at the node V_x and the home nodes of those queries q such as $u_{\mathcal{C}}(x) \in u_{\mathcal{C}}(q)$. Our evaluation study finds that this computation cost is small.

D. Un-Subscription

We propose that a subscription expires after a certain period. On expiration, it must be removed from the network. The removal of a subscription q also involves processing of some other queries. These queries are those stored at the home node of q because their projection is covered by the projection of q. For example, consider Figure 3(c). If query q_3 expires, not only it is removed from nodes 1, 2, 3, 4, 5, 6, 7, but also we need to replicate query q_1 at nodes 1, 2, 3, 4, 5. The procedure for removing a query when it expires is below.

Protocol 3.3 (Unsubscription Protocol):

1) *If the projection of query q does not cover any other subscription projection, remove q and quit*
2) *Else, for each query q' such that $u_{\mathcal{C}}(q) \supset u_{\mathcal{C}}(q')$*
    a) *If the projection of query q' is not covered by any other subscription projection, do nothing and quit*
    b) *Else, replicate q' according to the Subscription Protocol 3.1*

IV. Simulation Study

We evaluated the proposed solution based on a simulation. The sensor network by default was consisted of $n = 1000$ nodes located in a squre area $[0, 1]^2$. We simulated two different network configurations: one with the node locations uniformly distributed, and one with coverage holes. For each simulation run, $M = 10,000$ subscriptions were generated, each being a sphere with center chosen uniformly in random in $[0, 1]^d$. We considered various choices for the data dimension:

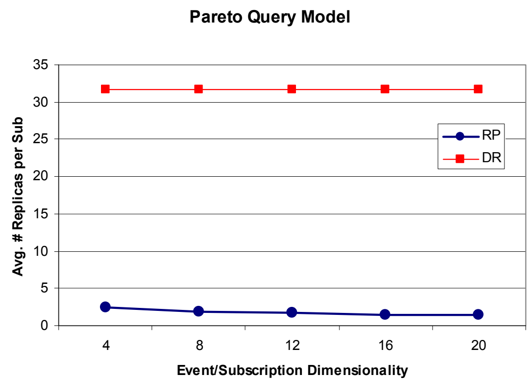


Fig. 4. Number of replicas per subscription

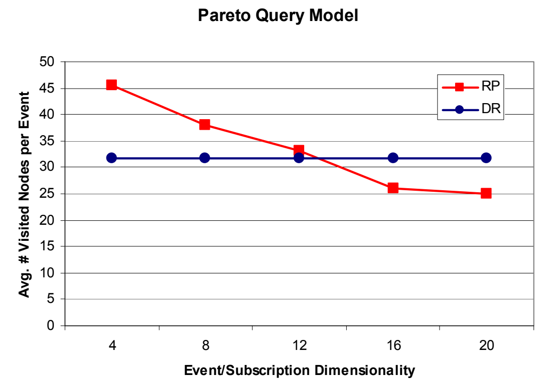


Fig. 5. Number of nodes visited per event

$d \in \{4, 8, 12, 16, 20\}$. The radii were chosen to a maximum of $r_{max} = 0.5$ and based on the Pareto 80/20 distribution. This model reflects the case typical in practice that most subscriptions are specific (i.e., small radius), only a few being extensive (i.e., large radius). 1,000 events were generated at nodes chosen equally likely in the network.

In this section, we report the results of the evaluation based on the following metrics:

- *Replication cost*: computed for each query, and represented by the number of nodes where this query is replicated.
- *Computation cost*: computed for each event, and represented by the number of nodes where we need to run the matching algorithm to find the queries matching this event.
- *Notification time*: computed for each event, and represented by the hopcount it takes the advertisement of this event to travel from its publishing node to the initiating node of each matching query.

These metrics were obtained as average values. We compared our solution (referred hereafter as RP - Random Projection) to Double-ruling [10]–[12] (referred to in the figures as DR). Double-ruling was chosen for our comparison because it is a popular technique that is the only existing to work with multidimensional data and range subscriptions. As discussed in Section 2, using horizontal/vertical lines results in less replication cost than using random walks or 3D great circles. We, therefore, evaluated against the horizontal/vertical-line version of double-ruling. We report the results in the following subsections (in *present* tense, for ease of expression).

A. Replication Cost

With Double-ruling, each subscription is replicated at all the nodes along a horizontal line, and, so, the average number of subscription replicas is roughly $\sqrt{n} = \sqrt{1000} \approx 32$, no matter the number of subscriptions. As shown in Figure 4, our solution offers a substantially less replication cost; no more than three nodes per subscription. A nice observation is that our replication cost is less as the dimensionality increases.

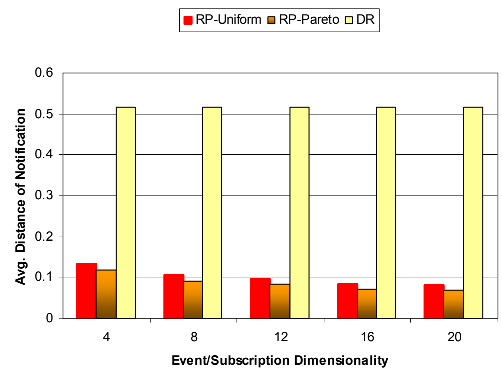


Fig. 6. Notification time

B. Computation Cost

By replicating subscriptions less, it is understandable that the advertisement of an event may be sent to more nodes to search for its matching subscriptions. Figure 5 illustrates this tradeoff, in which we compute the average number of nodes visited by each event that must run the algorithm to match subscriptions against it.

In Double-ruling, every node along the vertical line crossing the node detecting the event must run the matching algorithm. Thus, the number of nodes visited is always $\sqrt{n} \approx 32$, roughly.

We observe that RP results in more visited nodes when the data dimension is low. For example, when $d = 4$, an event visits an average of about 45 nodes, 15% more than that in Double-ruling. This number, however, is less than 5% of the network size and decreased as the data dimension gets higher. Especially, RP visits even fewer nodes than Double-ruling does when d is larger than 12. This study implies that our solution offers better notification efficiency when the data dimensionality is high .

C. Event Notification Time

The notification time is important in any mission-critical publish/subscribe system because we want to inform the subscribers as soon as an event occurs. We compute the average time, in terms of distance travelled, it takes an event to route to the nodes storing matching subscriptions. The results are shown in Figure 6 (where we also display the result for the case queries have uniformly distributed radii – denoted by RP-Uniform in the figure).

The average notification time for Double-ruling is approximately 0.5 no matter the dimensionality (because the advertisement of an event has to travel on average half the vertical line of the sensor area). Figure 6 apparently shows that RP is much quicker to notify upon an event, almost by a factor of five times compared to Double-ruling. This study implies that the event/subscription-locality property is well-preserved using our technique, as for each event we do not need to travel too far to find all the matching subscriptions.

D. Evaluation Remarks

Like Double-ruling, our solution does not suffer the curse of dimensionality problem. However, we offer much better efficiency, even when the network size is increased. In the aspects of replication cost, subscription load per node, notification time, RP is the clear better choice. It is also observed that as the dimension is increased, the costs are even reduced. This is explainable. According to Property (2) in Subsection III-A, the projection of a subscription is smaller when d is higher, thus covering fewer sensor nodes in the network. As a result, fewer replications are needed for each subscription (Figure 4), fewer nodes are visited per event (Figure 5), and less time is needed for the notification process (Figure 6).

A potential drawback of RP compared to Double-ruling is the higher number of nodes an event needs to visit to notify its matching subscribers (see Figure 5). Thus, for applications where a lot of events are generated, our technique might incur a large amount of event traffic. However, this drawback is less visible as the event dimension is higher.

V. Conclusions

We have proposed a cost-effective publish/subscribe design for sensor networks, that can work with range subscription queries and any dimensionality. The solution is based on a simple random projection into the sensor location space. The subscription load in the network is reduced by utilizing the covering relationship among the subscriptions in the projection space. Our evaluation study substantiates our findings, showing that the proposed design has a very low storage cost and high communication and computation efficiency. Its drawback is the notification delay when the data dimension is low. We would recommend our technique for applications where the subscriptions outnumber the events and where we are more concerned with the cost efficiency rather than the notification time.

Currently, we work with spherical subscriptions and our evaluation study assumes the uniform distribution for sensor locations. As the next step, we will address other types of subscriptions and evaluate sensor networks with non-uniform coverage. Investigation on the load balancing issue is also an item of our future work.

Acknowledgment

This work is funded in part by NSF grants CNS-0615055 and CNS-0753066, and by a UMass Boston FIPSE equipment grant. The authors are thankful for the support.

References

[1] E. Fidler, H.-A. Jacobsen, G. Li, and S. Mankovski, "The padres distributed publish/subscribe system," in *FIW*, 2005, pp. 12–30.

[2] L. Fiege, M. Cilia, G. Muhl, and A. Buchmann, "Publish-subscribe grows up: Support for management, visibility control, and heterogeneity," *IEEE Internet Computing*, vol. 10, no. 1, pp. 48–55, 2006.

[3] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf, "Design and evaluation of a wide-area event notification service," *ACM Transactions on Computer Systems*, vol. 19, no. 3, pp. 332–383, 2001. [Online]. Available: citeseer.ist.psu.edu/482106.html

[4] X. Li, Y. J. Kim, R. Govindan, and W. Hong, "Multi-dimensional range queries in sensor networks," in *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2003, pp. 63–75.

[5] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu, "Data-centric storage in sensornets with ght, a geographic hash table," *Mob. Netw. Appl.*, vol. 8, no. 4, pp. 427–442, 2003.

[6] B. Greenstein, D. Estrin, R. Govindan, S. Ratnasamy, and S. Shenker, "Difs: A distributed index for features in sensor networks," *Elsevier Journal of Ad Hoc Networks*, 2003. [Online]. Available: 043

[7] D. Ganesan, B. Greenstein, D. Estrin, J. Heidemann, and R. Govindan, "Multiresolution storage and search in sensor networks," *IEEE Trans. Storage*, vol. 1, no. 3, pp. 277–315, 2005.

[8] M. Demirbas and H. Ferhatosmanoglu, "Peer-to-peer spatial queries in sensor networks," in *Peer-to-Peer Computing*, 2003, pp. 32–39.

[9] Y.-C. Chung, I.-F. Su, and C. Lee, "Supporting multi-dimensional range query for sensor networks," in *ICDCS '07: Proceedings of the 27th International Conference on Distributed Computing Systems*. Washington, DC, USA: IEEE Computer Society, 2007, p. 35.

[10] Q. Fang, J. Gao, and L. J. Guibas, "Landmark-based information storage and retrieval in sensor networks," in *IEEE INFOCOM*, 2006.

[11] D. Braginsky and D. Estrin, "Rumor routing algorthim for sensor networks," in *WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. ACM, 2002, pp. 22–31.

[12] R. Sarkar, X. Zhu, and J. Gao, "Double rulings for information brokerage in sensor networks," in *MobiCom '06: Proceedings of the 12th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2006, pp. 286–297.

[13] D. A. Tran and T. Nguyen, "Subscription covering detection in publish/subscribe systems based on random projections," in *Proceedings of IEEE Int'l Conference on Collaborative Computing (COLLABORATE-COM 2007)*. White Plains, NY: IEEE Press, October 2007.

[14] ——, "Publish/subscribe service in can-based p2p networks: Dimension mismatch and the random projection approach," in *IEEE Conference on Computer Communications and Networks (ICCCN '08)*. Virgin Island, USA: IEEE Press, August 2008.

[15] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*. ACM Press, 2000, pp. 56–67. [Online]. Available: http://portal.acm.org/citation.cfm?id=345920

[16] Y. Huang and H. Garcia-Molina, "Publish/subscribe tree construction in wireless ad-hoc networks," in *MDM '03: Proceedings of the 4th International Conference on Mobile Data Management*. London, UK: Springer-Verlag, 2003, pp. 122–140.

[17] C. Avin and B. Krishnamachari, "The power of choice in random walks: An empirical study," *Comput. Networks*, vol. 52, no. 1, pp. 44–60, 2008.

[18] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger, "Geometric ad-hoc routing: of theory and practice," in *PODC '03: Proceedings of the twenty-second annual symposium on Principles of distributed computing*. New York, NY, USA: ACM, 2003, pp. 63–72.