Performance Analysis of Random Database Group Scheme for Mobility Management in Ad hoc Networks

Jiandong Li
State Key Lab of ISN
Xidian University
Xi'an, Shaanxi, 710071, China
jdli@mail.xidian.edu.cn

Zygmunt, J. Haas, and Ben Liang
School of Electrical and Computer Engineering
Cornell University
Ithaca,NY,14853 USA
{haas,liang}@ece.cornell.edu

***Abstract*-In this paper[*], the performance of a distributed mobility management scheme, the Randomized Database Group (RDG), for mobile ad hoc networks is presented. In this scheme, databases are used to store the location of the network nodes and to manage the mobility of nodes. When a mobile's location changes, a number of randomly selected databases are updated. When a mobile's location is needed, such as upon a call arrival, a number of randomly selected databases are queried. A number of different RDG query schemes are studied and their performance are compared. In particular, the optimum update-group size and the query-group size are found. We also present the probability of the first query being successful and the average query delay to find the mobile's location. Finally, we estimate the cost of implementing the RDG scheme as a function of different number of databases.**

Keywords-Mobility Management;Random Database Group; Ad hoc network

I. Introduction

Mobile ad hoc networks (MANETs) are an emerging technology that allows establishing an instant communication network for civilian and military applications [1]. Target applications range from collaborative, distributed mobile computing (sensors, conferences, classrooms, etc) to disaster recovery (such as fire, flood, earthquake), law enforcement (crowd control, search and rescue), and tactical communications (digital battlefield). A mobile ad hoc network is self-organizing network, and communications is carried out mostly through multihop routing over wireless links. Mobility of the network nodes, limited resources (e.g., bandwidth and battery power), and potentially large number of nodes make the routing and mobility management in ad hoc networks extremely challenging problems.

In mobile ad hoc networks, especially of large size, the cellular-like (HLR/VLR) mobile management scheme cannot be used, because ad hoc networks lack pre-existing infrastructure that is needed to facilitate such a scheme. In [2], a distributed mobile management scheme using a class of Uniform Quorum Systems (UQS) was proposed. In this scheme, mobiles' location information is stored in the network nodes themselves, which serve as an implementation of distributed location databases. The databases are dynamically organized into quorums, every two of which intersect at a constant number of databases. Upon location update or query, a mobile's location information is written to or read from all the databases of a quorum, which is chosen in a non-fixed manner.

In [3], another distributed mobility management scheme using Randomized Database Group (RDG) was proposed. Upon a location update or a query arrival, the mobile's location information is written to or read from a group of randomly chosen databases. The update and query operations of a mobile node's location are not necessarily to the same group of databases. Neither there is a guarantee that the two groups overlap. If they do, then the up-to-date location information can be returned to the querying node, by retrieving this information from the nodes that are on the intersection between the queried database group and the group that was last updated with the location information. The main motivation for this scheme is that although due to the dynamic nature of the network, databases may separate from the network for arbitrary long time periods, the location management scheme will still be able to provide the required information, as long as there are enough databases still present in the network. This may not be true for HLR-like schemes, where loss of some HLRs, even though small in number, may totally paralyze a part of network. Moreover, the RDG scheme is more flexible than UQS scheme.

In this paper, we analyze the cost of the RDG scheme and propose different update and query strategies. In Section II, we present an analytical model for RDG and we evaluate the average query times for a fixed query group size. In Section III, the dynamic performances of RDG with variable update rate and query rate are given. In Section IV, the probability that the first query is successful and the optimum update and query group sizes are calculated. In Section V, four different query strategies are proposed and compared. Section VI concludes the paper.

II. THE RDG NETWORK MODEL

A model of an ad hoc network with N nodes is shown in Fig. 1. Consider the information about a location of one of the network nodes. When the information becomes available, K databases are randomly chosen and the location information is updated (these are the "rectangular" nodes in the Figure).

[*] The work in this paper has been supported by TRAPOYT, 863 Project (2001AA123031), DVSP and Key project of MOE, China, by the AFOSR and the ONR, as part of the MURI program under the contracts numbers F49620-02-1-0217 and N00014-00-1-0564, respectively and by the NSF under the grant number ANI-0081357

When the location information of the node in question is requested (by any node in the network), *Q* databases are randomly selected and queried (these are the "oval" databases in the Figure). If the updated *K* databases and the queried *Q* databases have at least one common database (such databases are marked as both "rectangular" and "oval" in the Figure), the query will produce an answer and is considered successful. If this is not the case, a new set of *Q* databases is selected and queried. The process continues until in one of the rounds, a database, which contains the sought information, is found.

The operation of the RDG scheme can be modeled as “Sampling from dichotomous populations”[5], which states as follows: Consider a population, which consists of N balls numbered *1, 2, ..., N.* Suppose that *K* balls, numbered *1* through *K*, are white and that the remaining balls, numbers *K+1* through *N*, are black. Samples of *Q* balls are successfully drawn randomly from this population, until one of the samples contain at least one white ball. What is the probability that the first sample of *Q* balls contain at least one white ball? What is the average number of samples that need to be drawn?

Of course, in our analogy, the ball population represents the nodes in the network; the white balls are the balls that were updated with the location information, while the drawn balls are those databases that are queried. Let X denotes the number of drawings required to obtain a white ball (an updated database). We assume that K is positive; in other words, that there actually are some white balls (updated database) in the population.

First, assume that *Q=1.* The probability function of X, the number of query rounds that is required to obtain an updated database, is given by [5]:

$$P(X=k)=\left(N-K\right)^{(k-1)}K/N^{(k)} \quad , k = 1, 2, \ldots \qquad (1)$$

where $N^{(k)} = N\,(N-1)\,\ldots\,(N-k+1),\ k\leq N.$ (2)

Then the expected number of query rounds is:

$$E(X) = (N+1) / (K+1) \qquad (3)$$

Let us label *Y* as the number of updated databases within a sample of *Q* databases (Q>1). Then the probability function of *Y* is given as [5]:

$$P(Y=k)=\binom{Q}{k}\frac{K^{(k)}(N-K)^{(Q-k)}}{N^{(Q)}}=\frac{\binom{K}{k}\binom{N-K}{Q-k}}{\binom{N}{Q}} \qquad (4)$$

$$\equiv P(k,Q,K,N),\quad k=0,1,\ldots,Q$$

We are interested in the number of query rounds required; i.e., the probability function of *X,* which we obtain from (4) as follows:

$$\begin{aligned}&P(X=1) = 1- P(0,Q,K,N)\\&P(X=2) = P(0,Q,K,N)(1-P(0,Q,K,N-Q)) \qquad (5)\\&P(X=3) = P(0,Q,K,N)P(0,Q,K,N-Q)(1-P(0,Q,M,N-2Q))\\&\qquad\ldots\ldots\end{aligned}$$

The average number of queries is now obtained by using (5):

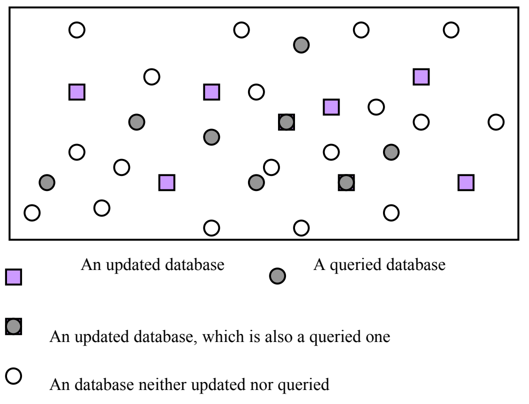
$$\begin{aligned}E(X)&=1\cdot P(X=1)+2\cdot P(X=2)+\ldots+(L+1)\cdot P(X=L+1)\\&=1+\sum_{l=1}^{L}\frac{(N-lQ)(N-lQ-1)\ldots(N-lQ-K+1)}{N(N-1)\ldots(N-K+1)}\ ,\end{aligned} \qquad (6)$$

where L is the integer part of *(N-K)/Q*.

When *Q=1*, equation (6) becomes, *E(X)=(N+1)/(K+1)*, which is consistent with equation (3), since:

$$\binom{n}{n}+\binom{n+1}{n}+\binom{n+2}{n}+\ldots+\binom{n+m}{n}=\binom{n+m+1}{n+1}$$

The average number of queries, *E(X)*, is shown in Fig. 2 for the case of *N=40.* (These values obtained by (6) match perfectly the simulation results.) As Q and K increase, the average number of queries converges to 1. The larger the updated group size K, the smaller the average number of queries.



An updated database
A queried database
An updated database, which is also a queried one
An database neither updated nor queried

Fig.1 Intersecting the update and the query sets

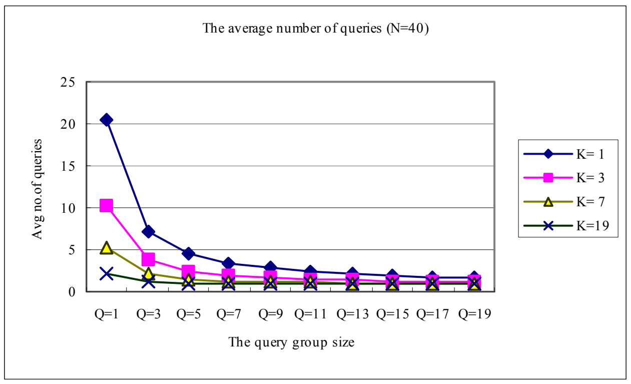


Figure 2: The average number of queries for N=40

We define the average query cost as: "*the size of queried database set (Q)*" times "*the average number of repetitions (E(X)) to find an updated database*" times " *the cost C_q per query*", i.e.,

$$C_Q = Q \bullet \left(1 + \frac{\sum_{l=1}^{L} \binom{N-lQ}{K}}{\binom{N}{K}} \right) \bullet C_q \quad , \qquad (7)$$

which is shown in Fig. 3 for $N= 40$, (C_q was assumed to equal one cost unit). From these figures, we deduct that the average cost for $Q=1$ is always the lowest.

III. Dynamic Performance Of RDG

The results obtained in section II are for a static scenario, i.e., there is a single update followed by the querying process. Of course, in a practical mobile network, the network nodes are continuously moving, creating streams of update and query messages.

To evaluate the cost of such a dynamic process, we have to pose assumptions about the update and the query processes. In particular, we assume that the update rate is λ_u (arrivals per unit time) and that the cost per update is C_u. We further assume that the querying rate is λ_q (arrivals per unit time) and that the cost per query is C_q . Under these assumptions, we have: the total update cost is $\lambda_u K\ C_u$ and the total query cost is $\lambda_q\ Q\ E(X)\ C_q = \lambda_q C_Q\ C_q$..The total cost for the update and the query operations is:

$$C_{total}=\lambda_u K C_u + \lambda_q Q E(X) C_q = \lambda_u (C_u K + (\lambda_q/\lambda_u) C_q Q E(X)) \qquad (8)$$

From Equation (8) and numerical results, we learn that the minimum total cost is dominated by the average query cost, and the contribution of the update cost to the minimum total cost can be neglected for $\lambda_q/\lambda_u>4$ and $K<5$, and that the minimum total cost is dominated by the update cost, and the contribution of the query cost to the minimum total cost can be neglected for $\lambda_q/\lambda_u<4$ and $K>9$. The total cost asymptotically tends to be linear in K for $\lambda_q/\lambda_u<4$.

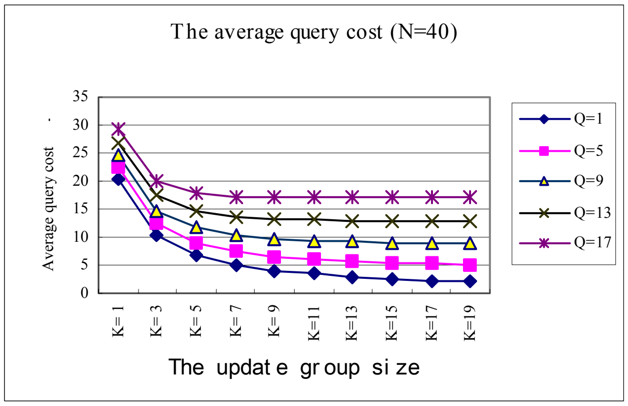


Fig 3: The average query cost for N=40

The minimum total cost for different $q= \lambda_q / \lambda_u$ and $N=40$ is shown in Fig. 4. From the figure, it is clear that the update rate dominates the minimum cost. If the network is highly dynamic, the update rate should be set to the same order of magnitude of the query rate to decrease the mobility management cost.

IV. OPTIMIZING THE COST AS A FUNCTION OF THE SUCCESSFUL PROBABILITY OF THE FIRST QUERY

The discussion in section III focused on the total average cost. In this section, we evaluate the optimum values for K and Q, so that, with high probability, the first query is successful.

The probability that the first query will be successful (i.e., the probability that on the first query we get at least one database that contains the location information) is equal to one minus the probability that no such databases are found in the first trial:

$$P(X=1) = 1 - \binom{N-K}{Q} \Big/ \binom{N}{Q} \qquad (9)$$

From the equation, we can find that although the total cost for small Q is low, the successful probability of the first query for small Q is quite low as well. If we want to maintain $P(X=1)$ in the range 70%-95%, the optimum values for K and Q are shown in Fig. 5 for $N=20,40,80,$ and 160, respectively and with $\lambda_q = \lambda_u =1$. The minimum cost for different P(X=1) is shown in Fig.6 (Strategy E).

From those figures, we conclude that for $P(X=1)\in (70\%; 90\%)$ and for $N=20,40, 80,$ and $160,$ the minimum cost is 9.07~9.69, 14.99~20.33, 22.01~28.67, and 32.4~42.82, the optimum K is 4~5, 8~10, 10~16, and 17~25, and the optimum Q is 2~3, 5~9, 8~12, and 11~17, respectively.

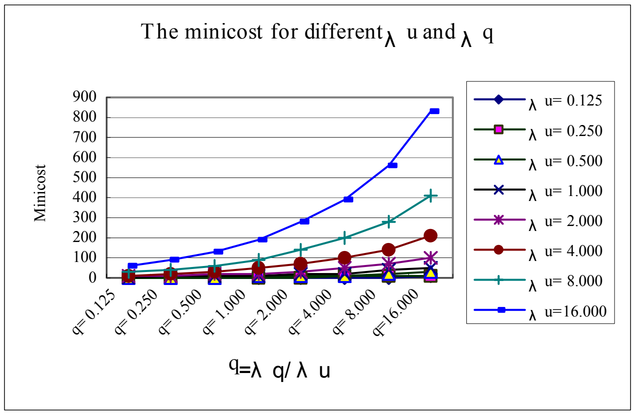


Fig. 4: The minimum total cost for N=40 and different $q \equiv \lambda_q / \lambda_u$

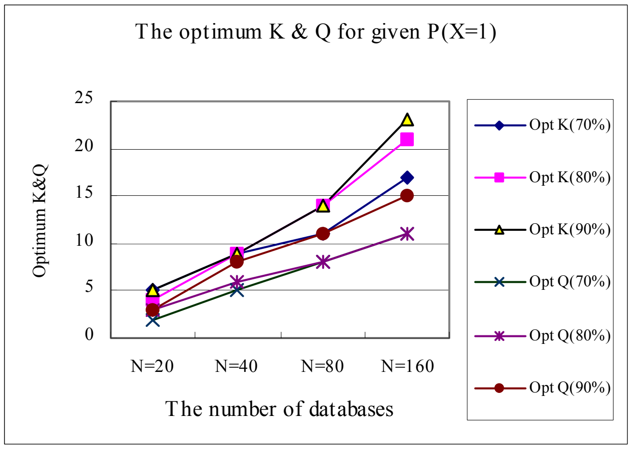


Fig. 5 Summary of the optimal values of K and Q for different N

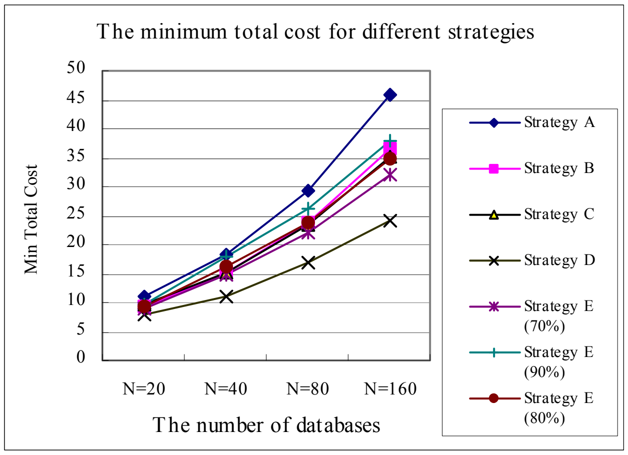


Fig. 6 The cost comparison of the different search strategy

V. Minimizing The Total Cost Subject To Maximum Search Delay

In the previous sections, we used a fixed query group size, and continue the trials until we find the updated database. In many applications, the delay in obtaining the location information is critical. For example, in supporting a real time application, small number of queries, and thus low delay, is desirable. In many cases, bounding, rather than minimizing, the number of queries suffices. In such cases, the goal is to minimize the cost, subject to maximum delay. In what follows, we present four possible searching strategies for the above problem.

- *Strategy A*: The first query uses a group size of *Q*; if the query is not successful, the second query uses a group size of *N-K-Q+1*. In this case, the maximum delay is two queries.
- *Strategy* B: The first *n* queries use the same query group size of *Q;* if none is successful, the final query uses a group size of *N-K-n*Q+1*. In this, we need at most n+1 queries. (This is a generalization of the *Strategy A*).
- *Strategy C*: The query group size is progressively increased; for example, the query group size could follow the sequence: *Q, 2Q, 3Q, …, (N-K-n*(n+1)*Q/2+1).*
- *Strategy D*: The lowest-cost query group size of *Q=1*is used.

The costs of the querying process for the above different strategies are as follow.

- *Strategy A*: $C_A=P(X=1)*Q+(1-P(X=1))*(N-K-Q+1)$ (10)
- *Strategy B*:

$$C_B=\sum_{l=1}^{n}P(X=l)*l*Q+(1-\sum_{l=1}^{n}P(X=l))(N-K-nQ+1) \quad (11)$$

- *Strategy C*:

$$C_C=\sum_{l=1}^{n}P_C(X=l)*(l*(l+1)/2)Q+(1-\sum_{l=1}^{n}P_C(X=l))(N-K-\frac{n(1+n)}{2}*Q+1)$$

(12)

,where $P_C(X=l)=P(0,Q,K,N)P(0,2*Q,K,N-Q)\ldots P(0,(l-1)*Q,K,N-(l-1)*(l-2)*Q/2)(1-P(0,l*Q,M,N-l*(l-1)*Q/2))$

- *Strategy D*: $C_D=(N+1)/(K+1),$ (13)

The comparison among the four cases is presented in Figures 6, 7, and 8. We assume here that *n=2* and that the query and update rates are equal. Figure 6 shows the minimum cost, while the optimal choices of the *K* and *Q* parameters are shown in Fig. 7 and 8. From those figures, it is clear that the *Strategy A* has the highest cost, while the *Strategy D* has the lowest cost. The cost of *Strategies B* and *C* are approximately equal. Of course, the gain in the cost is at the expense of delay: while *Strategy A* needs at most two queries, *Strategies B* and *C* need at most 3 queries, and *Strategy D,* which experiences the longest delay, needs on the average *(N+1)/2* queries. In general, to obtain low delay, large K and Q should be used.

Define *Strategy* E as one that use Q and K based on the results of equation (9) (Fig.5).

In Fig.6, the costs of *Strategy* A, B, C, and D are also compared with the cost of the *strategy* E. *Strategy* E tries to maximize successful probability of the first query.

If the successful probability of the first query needs to be 70%-80%, the cost of *strategy* E is between the costs of *Strategy* B and C. If the successful probability of the first query is set around 90%, the cost of S*trategy* E is lying between the costs of S*trategy* A and B.

(

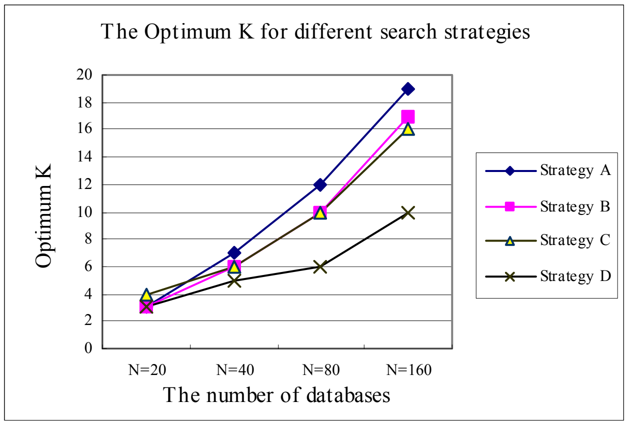


Fig. 7 The optimum K for the different search strategies

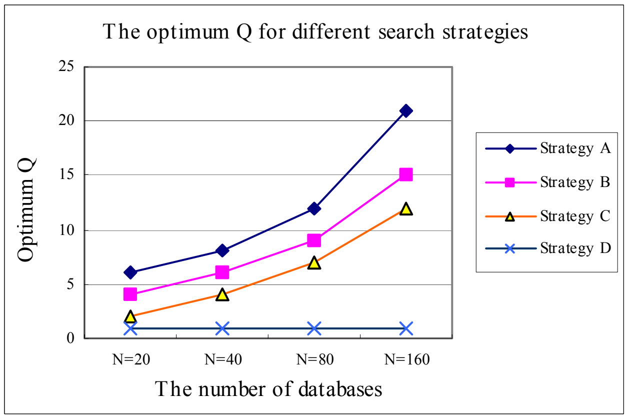


Fig. 8 The optimum *Q* for the different search strategies

VI. CONCLUSIONS

Random Database Group (RDG) is a simple but robust and efficient method for implementation of ad hoc mobility management. In this paper, we have presented an analytical RDG model, which allows us to calculate the cost (as the number of query messages) and the delay (as the number of querying rounds). As in any mobility management scheme, there is a tradeoff between the cost and the delay.

In general, if query cost is of more concern, a small query group size (*Q=1*, in the limit) is preferred. And conversely, if the delay is of concern, large query group should be used. The value of the update group size, *K,* depends, in general, on the values of five parameters: the size of the network (i.e., number of databases, *N),* the query rate (λq), the update rate (λu), the cost of a single query message (Cq), and the cost of a single update message (Cu). We have shown that most often there is an optimal value for *K,* which minimizes the cost.

Another intuitive conclusion that we can draw from our study is that as the environment becomes more dynamic, the cost of the update process has more effect on the total cost than the cost of the query process.

We have also studied the performance of the scheme as a function of the probability of the first query being successful, and we have evaluated the optimal values for *Q* and *K* that minimizes the total cost. This led us to define a number of search strategies for minimizing the cost, subject to delay constraints.

REFRENCES

[1] Xiaoyan Hong *et al*, "Scalable Ad Hoc Routing in Large, Dense Wireless Networks Using Clustering and Landmarks," in Proceedings of *2002 IEEE International Conference on Communications (ICC'2002),28 April-2May 2002, New York, NY USA*

[2] Zygmunt J. Haas and Ben Liang, "*Ad Hoc* Mobility Management with Uniform Quorum Systems, *IEEE/ACM Transactions on Networking,* Vol.7, Issue 2, April 1999,pp. 228 -240.

[3] Zygmunt J. Haas and Ben Liang, "Ad-Hoc Mobility Management with Randomized Database Groups," *1999 IEEE International Conference on Communications (ICC '99),* 6-10 June 1999 Vancouver, BC, Canada

[4] Ben Liang and Zygmunt J. Haas, "Virtual Backbone Generation and Maintenance in Ad Hoc Network Mobility Management," Proceedings of *INFOCOM 2000, Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies,* 26-30 March 2000 Tel Aviv, Israel

[5] Meyer Dwass, Probability: Theory and Applications, W.A . Benjamin, Inc., New York, 1970