Ad-Hoc Mobility Management with Randomized Database Groups

Zygmunt J. Haas and Ben Liang
323 Frank Rhodes Hall, School of Electrical Engineering
Cornell University, Ithaca, NY 14853
tel: (607) 255-3454, fax: (607) 255-9072
email: haas@ee.cornell.edu, liang@ee.cornell.edu

Abstract—**A distributed mobility management scheme using Randomized Database Groups (RDG) is proposed and analyzed for ad-hoc networks. In the proposed scheme, location databases are stored in the network nodes, comprising a virtual backbone within the flat network architecture. Upon location update or call arrival, a mobile's location information is written to or read from, respectively, a group of randomly chosen databases. Compared with a centralized scheme (such as the Home Location Register) with fixed associations, this scheme is more suitable for ad-hoc networks, where the connectivity of the nodes with the rest of the network can be intermittent and sporadic, and the databases are relatively unstable. The expected cost due to call loss and location updates using this scheme is analyzed in the presence of database disconnections. Based on the expected cost, we present the numerical determination and approximation of the optimal total location database number, the optimal database access group size, and the optimal location update frequency, under different network stability, traffic, and mobility conditions. Numerical results show that the RDG scheme provides an robust and efficient approach to ad-hoc mobility management.**

I. Introduction

In the ad-hoc network architecture, there is no pre-existing fixed network infrastructure. Nodes of an ad-hoc network are mobile hosts with similar transmission power and computation capabilities. Direct communication between any two nodes is allowed when adequate radio propagation conditions and network channel assignment exist. Otherwise the nodes communicate through multi-hop routing. The location of a mobile host must be identified before a call to the mobile host can be established.

Most of the proposed and existing systems directly send data packets to a destination node through pre-determined routes, without using any specialized databases to store the mobile nodes' location. To achieve this, the initiating node must either already have an up-to-date routing table to all the nodes in the network (pro-active routing) or try to determine the route on demand (reactive) [1]-[3], or as more recently proposed, a combination of both [4]. For a large network with many nodes, direct routing potentially poses very high traffic and computational demands. Multi-level ad-hoc routing schemes [5]-[7] with similarity to the cellular wireline-wireless hierarchy were also proposed, in which all packets are sent from the initiating node to the destination through a set of *backbone* nodes, which comprise a centralized subnet. Since every packet within the network must go through the subnet, these schemes impose very high demand of channel bandwidth and node stability on the backbone.

In [8], we propose and analyze an ad-hoc mobility management scheme that utilizes location databases that form a *virtual backbone*, which is dynamically formed and distributed among the network nodes. These databases serve only as containers for location storage and retrieval. Routing is carried out in the flat network structure involving every nodes in the network. That is, the routes do not necessarily go through the databases. The databases are organized into a Uniform Quorum System (UQS), consisting of quorums every two of which intersect at a constant number of databases. Upon location update or call arrival, a mobile's location information is written to or read from all databases in a quorum, chosen in a non-deterministic fashion. A dynamic and distributed "adaptive HLR" scheme is also analyzed in [8], and shown to be a limiting case of the UQS and suboptimal in general.

In this paper, based on the virtual backbone architecture, we propose a scheme with Randomized Database Groups (RDG). Like ad-hoc mobility management UQS, this scheme is also doubly distributed in the sense that both the database assignment and the database access are dynamic and non-deterministic. The assignment of a location database to reside in any mobile host is flexible, contingent upon the network node stability and traffic and mobility patterns. During the location update of a mobile host or when a call arrives to a mobile host, location of the mobile can be written to or read from any randomly chosen group of k databases. The write and read operations are not necessarily to the same group of databases. The up-to-date destination mobile location is provided to the source mobile host by the databases at the intersection between the queried database group and the group last written to by the destination mobile host.

At any instant, more than k location databases in an ad-hoc network may be separated from the network. However, due to the dynamic nature of the mobile and database association in this scheme, as long as some databases remain, location updating and location querying are still possible in the entire network. This is not true for the HLR-like schemes, where loss of some HLR-s, even though small in number, may totally paralyze the part of network that relies on those HLR-s. Thus, the distribution of responsibility among the location databases, which are themselves distributed among the mobile hosts, is the key to our scheme, which provides high degree of reliability in mobility management of ad-hoc networks. In fact, it can be shown that the "adaptive HLR" scheme is also a limiting case of the RDG scheme and is suboptimal in general.

Whether to use UQS or RDG depends on network node stability and system Quality of Service (QoS) requirement. The UQS scheme allows constant number of overlapped databases between the queried group and the updated group, hence providing more predictable QoS for each independent call initiation. However, because of the stringent requirements in the construction of UQS, the scheme is not appropriate when the average duration of database detachment is large. The RDG scheme, on the other hand, is very robust against such instability, and is relatively easy to implement.

From an experimental statistical point of view, the RDG can also be considered as a special case of the *tactical configuration*[9], or *t-designs*[10], [11], which is an extension of the Balanced Incomplete Block Design. The RDG construction can also be considered as a special case of the *probabilistic quorum systems* presented in [12], where each k-group is corresponding to one of the probabilistic quorums. It is probabilistic in the sense that every two groups intersect at a certain number of databases only with some probability.

In Section II, we describe in more detail the Randomized Database Group mobility management scheme. In Section III, we present a framework for evaluating the effectiveness of the RDG scheme in combating ad-hoc network database instability. In Section IV, we describe how to determine, numerically, the joint optimization of the number of required location databases, the group size, and the mobile location update frequency, under different network node stability, traffic and mobility patterns, and the relative cost of call loss and location update.

II. THE RDG MOBILITY MANAGEMENT SCHEME

A. Randomized Database Groups

In order to implement the mobility management scheme, a set of mobile hosts is chosen to contain the location databases. This set comprises a self-organizing *virtual backbone*. Please refer to [8] for detailed descriptions of the formation and maintenance of the virtual backbone in the presence of node disconnections. We say that a database *fails* when the node in which it resides detaches from the network for a long time, such that the location information stored in it is lost. We say that a database is *inaccessible* when it retains the location information but cannot be accessed for a short period of time, due to node instabilities.

Given a virtual backbone with n location databases, and a group size k, any combination of k databases forms a Randomized Database Group, where the location information of a mobile is stored and retrieved. When a mobile needs to update its location information, it uses any accessible RDG. Since there are C_k^n, where C denotes combination, database groups, the probability of not finding a complete group is small. Nevertheless, if there are less than k accessible database in the network, all of the remaining ones are used. A source mobile similarly queries k randomly chosen accessible databases for the location information of the destination mobile.

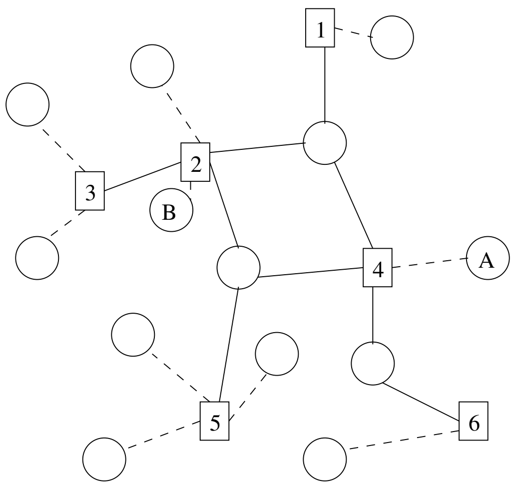

Fig. 1. Virtual Backbone and the Location Databases

Note here that k could be different for different mobile nodes. It can be made adaptive to each mobile node's traffic and mobility patterns. We assume that the database group size associated with each mobile is available at each virtual backbone node.

For Figure 1, there are $n = 6$ databases. Assuming that all databases are accessible, an example of RDG would be all combinations of size $k = 3$: $\{\{1, 2, 3\}, \{1, 2, 4\}, \{1, 2, 5\}, \ldots,$ etc$\}$. When a mobile host, A for example, updates its location, it contacts the closest node that belongs to the backbone set[1], the one with location database 4, in this case. The backbone node then sends out a location update message, containing A's ID number, the node's location (e. g., identity of the database 4), and a sequence number[2] that indicates the time of update, to one of the database groups, say, $\{1, 4, 5\}$. When a mobile host, B, initiates a call to A, it first contacts the node with location database 2, which in turn queries, say, $\{1, 2, 3\}$, for A's location. Location database 1 then sends back to B the location information of A. It is possible that location database 2 has an older copy of A's location information and sends it to B as well. Then, it is up to B to check the time sequence numbers in both messages and pick up the latest one.

With appropriately chosen k, the probability of non-intersection can be made sufficiently small. Therefore, with high probability, the location information of the destination mobile is available to the source mobile through the shared databases between the queried group and the updated group.

B. Mobile Location Updates

Another way of combating frequent location database failures is to restore the location databases through updating by the mobile hosts. There are three ways in which locations are

[1] A mobile host learns about its closest location database through algorithms such as the Zone Routing Protocol proposed in [4].

[2] The sequence numbers are generated by a counter built into to the mobile host. The counter is increased by one upon every location update by the mobile host, and is reset to zero when the maximum value is reached.

updated in the databases:
1. Call-origination update: When a mobile host initiates a call, it queries an RDG for the location of the destination and, at the same time, writes its current location into the queried databases. We model the length of time between any two consecutive call originations as a random variable, identically distributed with the probability density function(PDF) $f_o(t)$.
2. Location-change update: When a mobile host changes its location (in our example this constitutes changes in its closest location database), it updates its new location in an RDG. We also model the length of the time between any two consecutive location-change updates as a random variable with the PDF $f_m(t)$.
3. Periodic update: In order to avoid call loss during long time of lack of activity and immobility, a mobile host sends its location information to one of the RDGs periodically at every T_p units of time.

III. Mobility Management Cost of the RDG Scheme

In the RDG scheme, the destination mobile cannot be found, and hence a call is lost, if there is no overlapped location database between the database group updated by the destination mobile and the database group queried by the call initiating mobile, or if all the overlapped location databases have failed within the time interval from the update to the query. The sum of the penalties due to call loss and the cost of location updates and queries is an optimization function in our work. The minimum cost can be achieved by appropriate selection of the location update frequency and the RDG size.

Since each database is inaccessible with probability p_e, at any time instant, the PDF of the number of remaining accessible databases in the network is

$$p_{rem}(i) = \mathrm{C}_i^n (1-p_e)^i p_e^{n-i} , \qquad 0 \le i \le n . \tag{1}$$

Since a location update and a location query are initiated by different mobiles at different time instances, we assume that p_{rem} during these two processes ia independent. Therefore, the joint PDF of i accessible databases during an update and j accessible databases during a query is $p_{rem}(i)p_{rem}(j)$.

If there are more than k accessible databases, only k of them are used by a mobile. Therefore, the probability that r databases among the queried ones contain location information stored during the last update is

$$\begin{aligned} g_r(r) \;=\; & \sum_{i=0}^{k}\sum_{j=0}^{k} p_{rem}(i)p_{rem}(j)p_{ij}(r) \\ & + \sum_{i=0}^{k}\sum_{j=k+1}^{n} p_{rem}(i)p_{rem}(j)p_{ik}(r) \\ & + \sum_{i=k+1}^{n}\sum_{j=0}^{k} p_{rem}(i)p_{rem}(j)p_{kj}(r) \\ & + \sum_{i=k+1}^{n}\sum_{j=k+1}^{n} p_{rem}(i)p_{rem}(j)p_{kk}(r) , \end{aligned} \tag{2}$$

where $p_{ij}(r)$ is the probability density of the intersection size r given i updated databases and j queried databases, and $p_{ij}(r) = \mathrm{C}_r^i \mathrm{C}_{j-r}^{n-i}/\mathrm{C}_j^n$, for $\max(0, i+j-n) \le r \le \min(i,j)$, and $p_{ij}(r) = 0$, otherwise.

We define t_r as the time interval between the arrival of a call and the immediately preceding update event. Then

$$t_r = \min(t_o, t_m, t_p) , \tag{3}$$

where t_o, t_m, and t_p are the time intervals between the call arrival and the last call-origination update, the last location-change update, and the last periodic update, respectively.

Consider a call arrival as a random incidence upon the stochastic processes consisting of the above three types of random update intervals. t_o has the density function

$$g_o(t) = \frac{1 - F_o(t)}{\int_0^\infty t f_o(t)dt} , \tag{4}$$

t_m has the density function

$$g_m(t) = \frac{1 - F_m(t)}{\int_0^\infty t f_m(t)dt} , \tag{5}$$

and t_p has the density function

$$g_p(t) = \frac{1}{T_p} . \tag{6}$$

Then the density function of t_r can be computed by

$$\begin{aligned} f_r(t) = g_o(t)[1-G_m(t)][1-G_p(t)] + g_m(t)[1-G_o(t)] \times \\ [1-G_p(t)] + g_p(t)[1-G_o(t)][1-G_m(t)] . \end{aligned} \tag{7}$$

If there are r shared databases between the queried and updated RDGs, a call is lost only if all these r databases have failed during the time interval between the call arrival and the last location update. We assume that the database failures have Poisson distribution with mean time between failures T_f. The probability that one database fails within t_r is $1-e^{-\frac{t_r}{T_f}}$. Hence, given r, the expected number of lost calls per unit time is

$$\begin{aligned} E_{loss} \mid r \;&=\; \lambda_a \Pr\{r \text{ databases fail within } t_r\} \\ &=\; \lambda_a \int_0^{T_p} \left(1 - e^{-\frac{t}{T_f}}\right)^r f_r(t)dt . \end{aligned} \tag{8}$$

Therefore, the expected penalty per unit time due to call loss is

$$C_{loss} = c_l \lambda_a \sum_{r=0}^{k} g_r(r) \int_0^{T_p} \left(1 - e^{-\frac{t}{T_f}}\right)^r f_r(t)dt . \tag{9}$$

In the general case of routing in ad-hoc networks, the cost of a mobile host accessing a database can be estimated as the distance, in hops, from the mobile host to the database. Assuming that the databases are distributed uniformly among fixed total number of mobile hosts throughout a fixed network system coverage, the expected cost of accessing one database, c_u, has constant value depending only on the size of the coverage area.

We emphasize that, during the location update or location query process, a mobile accesses k *randomly chosen* databases (except maybe the nearest one). Although to access the k *nearest* databases reduces the cost of updates, the queried RDG and the updated RDG hardly intersects if the location updating mobile and the call initiating mobile are far apart. Therefore, for group size k, the expected cost of accessing an RDG is kc_u, independent of the parameters of the traffic and the mobility patterns. The cost per unit time due to location updates is

$$C_{update} = kc_u \left(\frac{1}{T_p} + \frac{1}{\int_0^{\infty} t f_o(t) dt} + \frac{1}{\int_0^{\infty} t f_m(t) dt} \right) . \tag{10}$$

The sum of the costs due to lost calls and due to updating location databases gives the total cost function[3]. Given the virtual backbone node stability, the cost of call loss, the average cost of database access, and the user traffic and mobility patterns, the mobility management cost is a function of n, k, and T_p:

$$C_{total}(n, k, T_p) = C_{loss} + C_{update} . \tag{11}$$

IV. Optimal RDG Determination through Cost Analysis

In the following numerical analysis, we assume that a mobile's call origination can be modeled by a Poisson point process, and

$$f_o(t) = \lambda_o e^{-\lambda_o t} , \tag{12}$$

where λ_o is the rate of call origination. We also assume that a mobile's location-change update is memoryless, and

$$f_m(t) = \lambda_m e^{-\lambda_m t} , \tag{13}$$

where λ_m is the rate of location change. These assumptions appear reasonable to many practical ad-hoc networks.

We will normalize C_{total}, so that it is expressed in units of c_u. We will also express time units in terms of $1/\lambda_a$.

For simplicity, we assume a totally symmetrical nodal traffic pattern, i. e., $\lambda_o = \lambda_a = 1$. Also, since location database resides in mobile hosts, and our definition of the mobile host location depends on the databases, we will assume that $T_f = 1/\lambda_m$. However, we emphasize that the above framework for cost analysis is applicable to systems without these assumptions as well.

A. Robustness of the RDG Mobility Management Scheme against Location Database Inaccessibility

In Figures 2 and 3, we fix $n = 32$, and let k take on the integer values from 4 to 32. p_e assumes values of 0, 0.2, 0.5, and 0.8. The other parameters are: $T_f = 1$ and $c_l = 1000$. For each data point, we use the bisection method to numerically find the

[3]Note that location queries can be performed together with call-origination location updates, hence not incurring any additional cost.

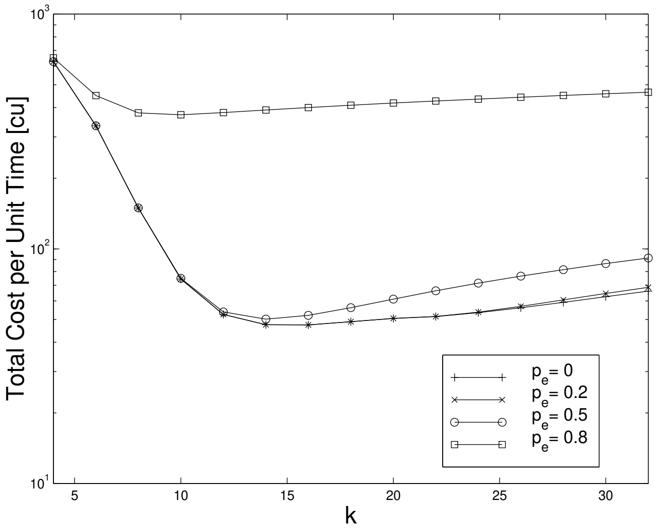

Fig. 2. Cost Optimization over T_p and k with Varying p_e

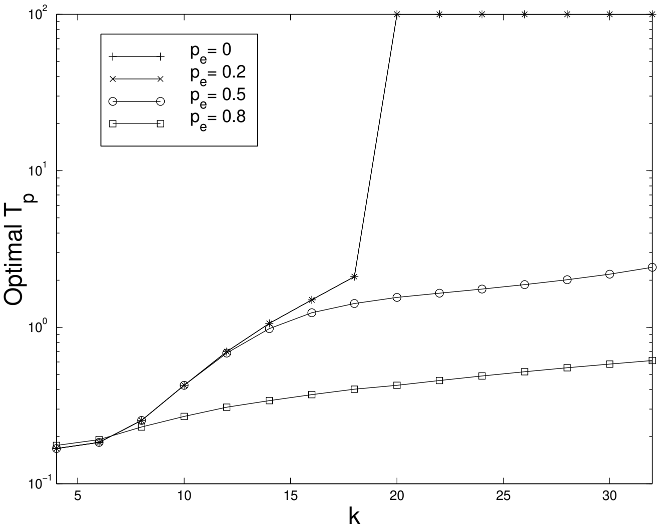

Fig. 3. Optimized T_p with Varying k and p_e

optimal value T_p^* and the minimized cost $C_{total}(T_p^*)$. A maximum value of $T_p = 100$ is assumed throughout this paper to facilitate the numerical computations. $T_p^* = 100$ means that the mobile periodic location update is not necessary.

Figure 2 shows that the RDG scheme is robust against the frequent detachments of nodes in an ad-hoc network. The curve for $p_e = 0$, representing the case where the location databases are always connected to the network, and the one for $p_e = 0.2$ are almost overlapping, suggesting that the mobility management cost is the same for these two cases. Even with $p_e = 0.5$, representing the case where the location databases are inaccessible half of the time, the cost remains very close to that under the ideal $p_e = 0$ case.

From Figure 3, we see that T_p^* increases as k increases. This matches with the intuition that, as k increases, the queried group and updated group intersection probability $g_r(r)$ shifts

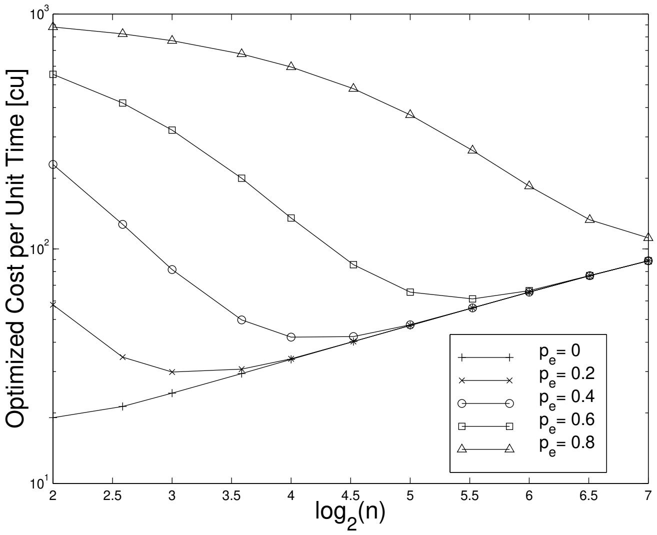

Fig. 4. Cost Optimization over n, k, and T_p with Varying p_e

such that larger r has higher probability, hence more protection against database failures is achieved, and less location updates are necessary. In particular, for both $p_e = 0$ and $p_e = 0.2$, when $k \geq 20$, the mobile should not perform periodic updates at all.

In Figure 4, we set $T_f = 1$, $c_l = 1000$, and allow n to vary from 2^2 to 2^7. For each value of p_e taken from $[0, 0.2, 0.5, 0.8]$, we obtain C_{total} jointly optimized over k and T_p. A two-level bisection method is used, with the optimal k obtained based on the optimization of T_p. An interesting observation here is that, for all values of p_e, the optimal $n = n^*$ has the property that

$$C_{total}(n^*, p_e) \approx C_{total}(n^*, 0) \ . \tag{14}$$

This suggests that the optimal number of location databases should be large enough such that the effect of some databases being inaccessible is minimal. This further suggests that the number of accessible databases at any time instant should be at least as large as k^*. Hence, for large n^*, we have

$$k^* \approx n^* \times (1 - p_e) \ . \tag{15}$$

We will confirm this observation in the more detailed analysis of the optimization of C^*_{total}, n^*, k^*, and T^*_p, to be presented in the next section.

B. The Effect of p_e, T_f, and c_l on Cost Optimization

The Figures 5-7 show the optimized C_{total}, n, k, and T_p vs. p_e, T_f, and c_l, respectively. The optimized values are obtained using a three-level bisection gradient descend method, with the optimal n computed based on the optimization of k and T_p, as described in the previous two sections. The default values are $p_e = 0.3$, $T_f = 1$, and $c_l = 1000$. The following ranges for the independent parameters are used: $p_e \in [0, 0.6]$, $T_f \in [0.1, 10]$, and $c_l \in [10, 10000]$. These ranges cover the typical parameter values of practical interest.

From part (a) of these figures, we see that C^*_{total} is a nearly linear function of p_e and T_f. Most interestingly, although C_{total} is a linear function of c_l, as shown in (11), C^*_{total} is a linear function of $\log(c_l)$, as shown in Figure 7(a). This suggests that the RDG mobility management scheme is applicable for a wide range of ad-hoc network stability, traffic, and mobility patterns, and it is particularly suitable for networks where the guarantee of call connection is heavily emphasized.

Part (b) of these figures show that n^* is approximately exponentially increasing with p_e. However, we also see that it is approximately linear with respect to $\log(T_f)$ and $\log(c_l)$. Thus, for typical parameter values, n^* is only in the order of tens of databases, which is achievable in a reasonably sized ad-hoc network. This small requirement of database quantity also lessen the computational burden on the generation and maintenance of the virtual backbone.

Part (c) of these figures clearly suggests that $k^*/n^* \approx 1-p_e$, as discussed in the previous section. The optimal k/n seems to be independent of T_f and c_l.

In particular, from Figure 5, we see that when the inaccessibility of the databases is small ($p_e < 0.05$), $k = n$, where n is in the single digits. Thus, in this case, we need a small amount of databases, but all databases are used for each location update or query. Note that this is exactly the "virtual HLR" scheme described in Section I. However, when p_e is larger, the required total number of databases increases exponentially, and the relative group size decreases linearly. For $p_e > 0.45$, the optimal k/n that gives the minimum cost is actually less than 0.5. In this RDG construction, two database groups do not necessarily intersect, but the system's ability to combat database inaccessibility favorably offsets the call loss penalty due to no group intersection.

Part (d) of these figures suggests that T^*_p should be in the order of T_f, with smaller T^*_p for larger c_l, and roughly, larger T^*_p for larger p_e. We also see that at some points where T_f is large or c_l is small, T^*_p jumps to 100, indicating that the mobile periodic updates are unnecessary beyond those points (e. g., for $p_e = 0.3$, $c_l = 1000$, and $T_f \leq 4$, or for $p_e = 0.3$, $T_f = 1$, and $c_l \leq 60$).

The above observations on k^* and T^*_p provide easy approximations to the optimal parameters. If precise values are preferred, these approximations can be used as the initial values for the gradient descend methods, reducing the convergence time and the amount of computation. This is especially useful in an ad-hoc environment where the node stability, traffic pattern, mobility pattern, or call connection priorities change over time, and the frequent re-computations of the adaptive system design parameters are desirable.

Finally, we notice that the oscillations on some of these curves are due to the integral nature of n and k selections.

V. Conclusions

In this paper, we propose the RDG scheme, where copies of the location information of a mobile can be stored and retrieved in a distributed fashion through the randomized access of database groups that are themselves randomly distributed among the network nodes. A framework is presented for evaluating the cost of mobility management using RDG under generic node traffic and mobility patterns. We investigate the effect of the network traffic and mobility patterns, location database stability, and the relative cost of call loss, on optimizing the database group constructions and the location update frequencies.

Under the assumption of some common network traffic and mobility patterns, our numerical analysis suggests that the RDG scheme is robust against node instability in ad-hoc networks. In particular, for $p_e \geq 0.1$, it outperforms the "adaptive HLR" scheme.

The optimal virtual backbone size can be made adaptive to each node's traffic and mobility patterns. Numerical results show that it is typically less than 100, which suggests that the implementation of the RDG scheme is a practical one. If, for reasons other than mobility management, a network requires more that n^* location databases, a partitioning scheme as proposed in [8] can be applied to achieve the optimal mobility management cost.

We have shown that the optimal group size can be approximated by a simple product of the optimal total number of databases and the database accessibility. We have also shown that the optimal mobile periodic location update period is on the order of the mean time between database failures. These properties, combining with (14), which gives an approximation of the optimal virtual backbone size, allow fast computation of the optimal parameters of the RDG scheme that achieve the minimal mobility management cost, adapting to the changing ad-hoc network environment.

REFERENCES

[1] J. J. Garcia-Luna-Aceves et al, "Analysis of Routing Strategies for Packet Radio Networks," *Proceedings of IEEE INFOCOM'85*, Washington, DC, March 1985.

[2] B. M. Leiner, D. L. Nielson, and F. A. Tobagi, "Issues in Packet Radio Network Design," *Proceedings of the IEEE*, vol. 75, pp. 6-20, January 1987.

[3] J. Jubin and J. D. Tornow, "The DARPA Packet Radio Network Protocols," *Proceedings of the IEEE, Special Issue on Packet Radio Networks*, vol. 75, pp. 21-32, January 1987.

[4] Z. J. Haas and M. R. Pearlman, "Providing Ad-Hoc Connectivity with the Reconfigurable Wireless Networks," *Proceedings of the ACM SIGCOMM'98*, September, 1998.

[5] A. Ephremides, J. E. Wieselthier, and D. J. Baker, "A Design Concept for Reliable Mobile Radio Networks with Frequency Hopping Signaling," *Proceedings of the IEEE*, vol.75, no.1, 1987.

[6] B. Das and V. Bharghavan, "Routing in Ad-Hoc Networks Using Minimum Connected Dominating Sets," *IEEE Int. Conf. on Communications*, June, 1997.

[7] J. Sharony, "A Mobile Radio Network Architecture with Dynamically Changing Topology Using Virtual Subnets," *MONET*, vol. 1, no. 1, pp. 75-86, 1996.

[8] Z. J. Haas and B. Liang, "Ad-Hoc Mobility Management With Uniform Quorum Systems," to appear on the *IEEE/ACM Transactions on Networks*.

[9] A. Dey, *Theory of Block Designs*, 1986.

[10] A. Hedayat and S. Kageyama, "The family of t-designs, Part I." *J. Statistical Planning and Inference*, 4, pp. 173-212, 1980.

[11] S. Kageyama and A. Hedayat, "The family of t-designs, Part II." *J. Statistical Planning and Inference*, 7, pp. 257-287, 1983.

[12] D. Malkhi, M. Reiter, and R. Wirght, "Probabilistic Quorum Systems," *Proc. ACM PODC*, pp. 267-273, Santa Barbara, 1997.

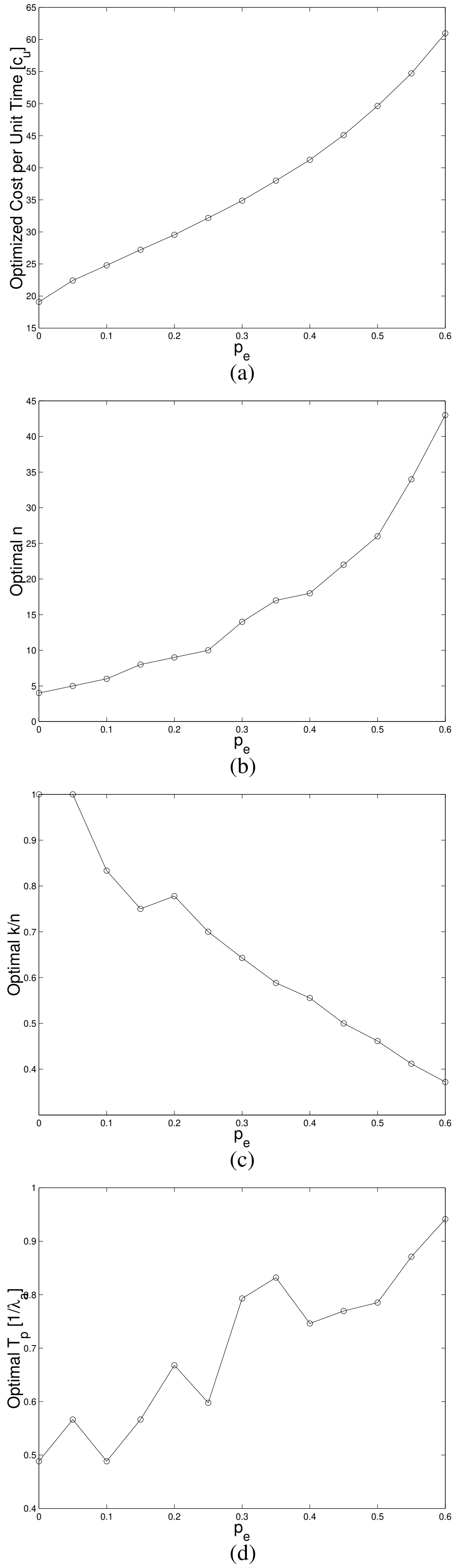

Fig. 5. Optimization vs. p_e: (a) C_{total}; (b) n; (c) k; (d) T_p.

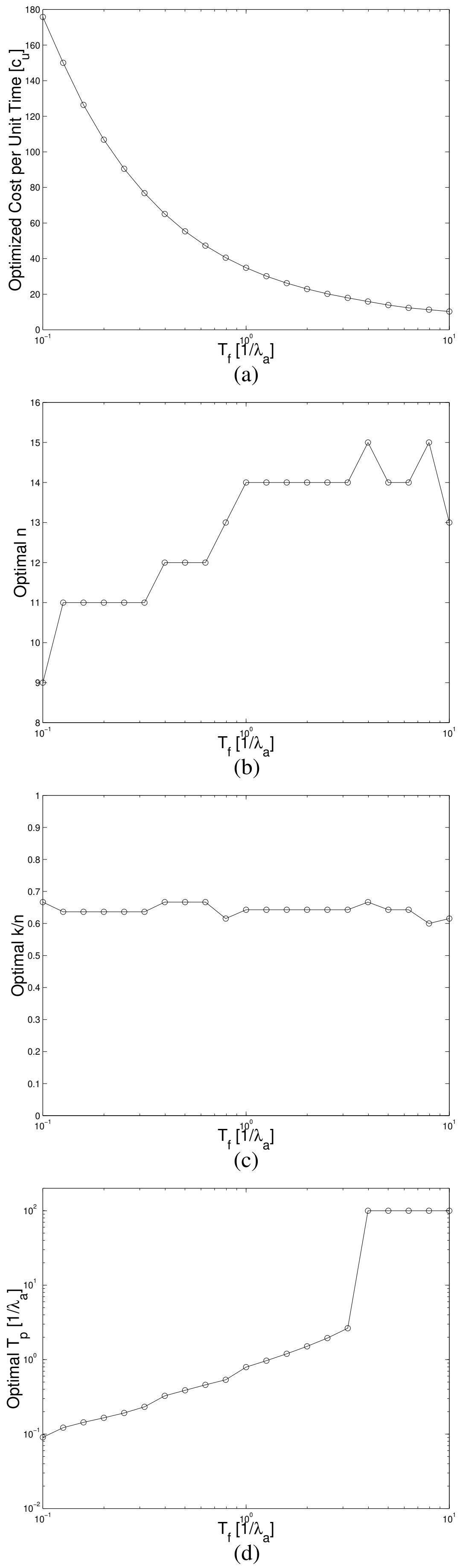

Fig. 6. Optimization vs. T_f: (a) C_{total}; (b) n; (c) k; (d) T_p.

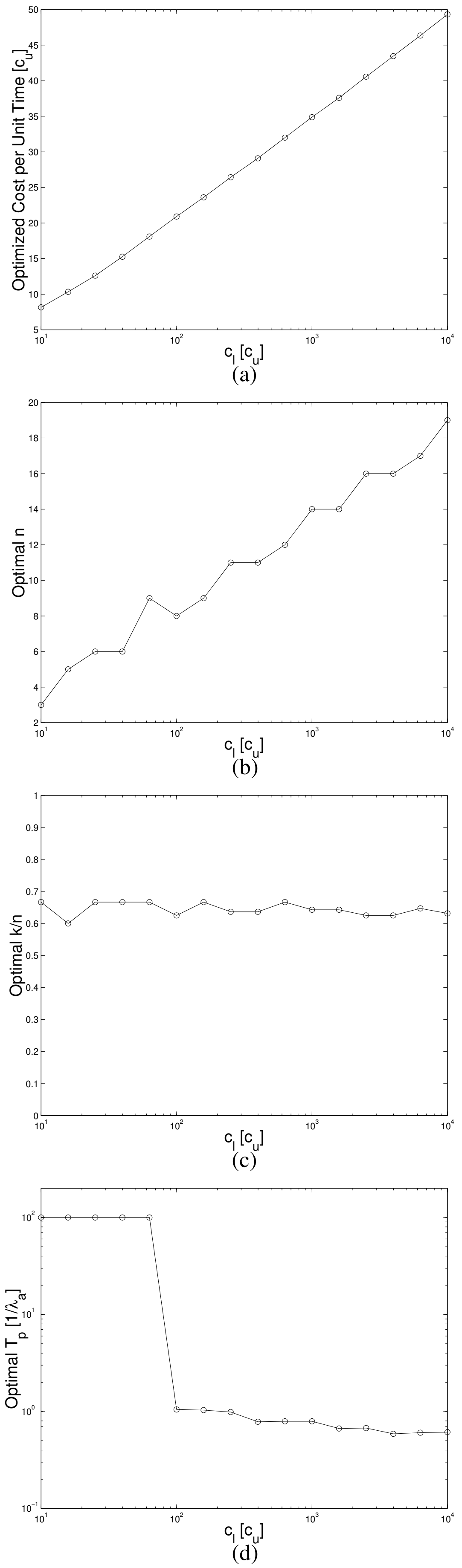

Fig. 7. Optimization vs. c_l: (a) C_{total}; (b) n; (c) k; (d) T_p.