Ad Hoc Mobility Management With Uniform Quorum Systems

Zygmunt J. Haas, *Senior Member, IEEE* and Ben Liang, *Student Member, IEEE*

Abstract— **A distributed mobility management scheme using a class of Uniform Quorum Systems (UQS) is proposed for ad-hoc networks. In the proposed scheme, location databases are stored in the network nodes themselves, which form a self-organizing virtual backbone within the flat network structure. The databases are dynamically organized into 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 the databases of a quorum, chosen in a non-deterministic manner. Compared with a conventional scheme (such as the use of Home Location Register (HLR)) 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. We introduce UQS, where the size of quorum intersection is a design parameter that can be tuned to adapt to the traffic and mobility patterns of the network nodes. We propose the construction of UQS through the Balanced Incomplete Block designs. The average cost due to call loss and location updates using such systems is analyzed in the presence of database disconnections. Based on the average cost, we investigate the trade-off between the system reliability and the cost of location updates in the UQS scheme. The problem of optimizing the quorum size under different network traffic and mobility patterns is treated numerically. A dynamic and distributed HLR scheme, as a limiting case of the UQS, is also analyzed and shown to be suboptimal in general. It is also shown that partitioning of the network is sometimes necessary to reduce the cost of mobility management.**

Keywords—***Ad hoc* network, balanced incomplete block design, data distribution, mobility management, quorum system, reconfigurable wireless network, uniform quorum system.**

NOMENCLATURE

λ_a	Arrival rate of calls to a mobile host.
λ_o	Arrival rate of call originations.
λ_c	Arrival rate of location changes.
λ_u	$\lambda_u = \lambda_o + \lambda_c$.
t_u	Time interval between a call arrival and the last call-origination or location-change update by a mobile host.
f_u	Density function of t_u.
T_p	Period of the mobile host periodic location update.
t_p	Time interval between a call arrival and the last periodic location update by a mobile host.
f_p	Density function of t_p.
t_r	$t_r = \min(t_p, t_u)$.
f_r	Density function of t_r.
T_f	Average time between two consecutive failures of a location database.
p_e	Steady-state probability of a database being inaccessible.
P_Q	Steady-state probability that at least one complete quorum exists in the system.
E_{loss}	Expected number of lost calls per unit time.
c_l	Expected cost of losing a call.
c_u	Expected cost of assessing one location database.
C_{total}	Cost function to be minimized.

Manuscript received March 10, 1998; revised October 15, 1998; approved by IEEE/ACM TRANSACTION ON NETWORKING Editor I. F. Akyildiz. This work was supported by the National Science Foundation under Grant ANI-980594.

Z. J. Haas is with the School of Electrical Engineering, Cornell University, Ithaca, NY 14850 USA (e-mail: haas@ee.cornell.edu).

B. Liang is with the School of Electrical Engineering, Cornell University, Ithaca, NY 14850 USA (e-mail: liang@ee.cornell.edu).

Publisher Item Identifier S 1063-6692(99)03633-X.

I. INTRODUCTION

IN A MOBILE communication network, mobile hosts move freely from place to place. The location of a mobile host must be identified before a call to the mobile host can be established. Mobility management deals with the storage, maintenance, and retrieval of the mobile host location information.

In the cellular construction, the system coverage area is partitioned into cells. Mobile hosts within a cell communicate with other terminals through a base-station installed in the cell and connected to the Public Switching Telephone Network. The location of a mobile host in cellular networks are defined in terms of the cells. The two commonly used standards, the Electronics Industry Association/Telecommunications Industry Association Interim Standard 41 in North America and the Global System for Mobile Communications in Europe, utilize home location registers (HLR) and visitor location registers (VLR), both residing within the wireline backbone, to keep track of user locations [1][2]. A mobile host's location information is recorded in its designated HLR/VLR databases through a *location update* procedure. When a call is made to the mobile host, its location information is retrieved from the HLR/VLR databases through a *location query* procedure.

In the ad-hoc network architecture, however, 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, in which case we say that these two nodes are neighbors. Otherwise the nodes communicate through multi-hop routing [3]. The lack of fixed infrastructure suggests that some network functions otherwise handled by the wireline backbone must now be maintained by the nomadic nodes in the ad-hoc network.

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) [4]-[6], or as more recently proposed, a combining of both [3]. For a large network with many nodes and frequent topology change, direct routing potentially poses very high traffic and computational demands. Multi-level ad-hoc routing schemes [7]-[9] 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 this work, we propose an ad-hoc mobility management scheme that utilizes location databases which form a *virtual backbone* that is dynamically 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 (see reference [3]) involving every nodes in the network. That is, the routes do not necessarily go through the databases. However, the node location provided by the databases can provide vital information to the routing protocol, so that route searching is more localized.

Moreover, the location of a mobile host is defined in terms of the positional relationships between the mobile host and the other nodes. The identity of the neighboring nodes of a mobile host can provide indication on how a message could be routed to the mobile host. In particular, we will define the location of a mobile host as the ID number of its nearest location database[1].

Nodes containing the location databases can dynamically detach and re-attach to the network at any time due to mobiles' movements or changes in the communications environment. We will see in Section II that a location database *fails* whenever the node containing the database detaches from the network for a long time. However, the ups and downs of a node's connectivity to the network should have minimum effect on the other nodes' communication. This imposes great challenges in the design and operation of an ad-hoc network. The traditional algorithms for mobility management, which rely on the base-stations and Mobile Switching Centers, are not applicable here. For example, in the HLR-like schemes, which mimic the cellular networks and set up fixed association between mobiles and databases, the databases would too often fail to provide the sought mobile's location due to frequent disconnections of the databases.

One obvious extension to the IS-41 and the GSM HLR/VLR schemes is to use multiple HLRs in ad-hoc networks. In such a case, unless all HLRs have failed (probability of such an event can be made small using large number of HLRs) the location information is available to the call initiating mobile host. However, such schemes still suffer from the disadvantages of fixed association between databases and the mobiles that store these locations.

An improvement to this scheme is to dynamically assign network nodes to perform the HLR function depending on the time-varying network node stability. When a node that serves as an HLR disconnects from the rest of the network, a new active node should be chosen to generate and maintain a new HLR with the same identity as the disconnected one. Such a "virtual HLR" or "adaptive HLR" scheme severs the association between a location database and the mobile host where the database resides. However, the association between a mobile and the databases where the mobile's location information is stored is still maintained.

Our proposed scheme of quorum systems is doubly distributed in the sense that it dynamically assigns network nodes to contain the location databases, and that it allows a mobile to access randomly chosen databases.

A quorum system is formed by organizing objects, databases in this case, into sets, called quorums, where every two quorums intersect and no quorum contains another quorum. The quorum system based scheme is distributed in operation and uses the databases of an entire quorum to hold replica of a mobile's location information. There is no fixed association between any particular quorum to any mobile host. First of all, 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. Secondly, 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 quorum of databases. (In this work we refer to the term "accessing a quorum" as "accessing all the databases in that quorum"). The write and read quorums are not necessarily the same. The up-to-date destination mobile location is provided to the source mobile host by the databases at the intersection between the read quorum and the quorum last written to by the destination mobile host. Two consecutive write or read operations performed by the same mobile may not access the same quorum either.

At any instant, some location databases in an ad-hoc network may be separated from the network. However, because of the dynamic nature of the mobile and database association in this scheme, as long as at least one quorum remains, 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 HLRs, even though small in number, can easily paralyze part of the network. Moreover, as we will show later in this paper, the "adaptive HLR" scheme is a limiting case of the quorum system scheme, and is in general inferior.

Thus, the distribution of responsibility among the quorums is the key to our scheme, which provides high degree of reliability in mobility management of ad-hoc networks. The other advantages of quorum systems in mobility management include location and service provider independent numbering [10][11], load balancing among location servers [10][11], and provision of system security [13][14].

Many quorum system construction schemes have been proposed in applications of data replication and distributed computation [10]-[12][15]-[19]. However, in these constructions, the minimum size of the quorum intersection is by design equal to one, therefore applicability of these schemes to our problem is limited.

In particular, [10] and [11] present a mobility management

[1]Other location indications are also possible. For example, mobile hosts equipped with the Global Positioning System can be deployed to provide grid-based location definition. The particular definition is irrelevant to the mobility management scheme presented here.

scheme for cellular networks using square quorum systems. The main goal of that work is to provide balanced database loading, since database failure is generally considered a rare event in the wireline networks.

An interesting region-based hierarchical overlapping read-write set mechanism for tracking mobile users is proposed in [20] based on a *match-making* theoretical account in [21]. However, the analysis in [20] is based on the assumption of pre-existing location directories, possibly residing in a centralized network entity, such as the wireline backbone in cellular networks. Furthermore, the directory operations are assumed to be free from failures. Therefore, this mechanism is not suitable for the volatile communication environment of ad-hoc networks. However, the notion of hierarchical overlapping sets does provide an interesting extension to the definition of quorum systems, which is not explore here. In this work, we consider the application of quorum systems (as they are regularly defined) to ad-hoc mobility management.

In some existing research works [19][22], the similarity between the quorum system construction and the much-studied Partially Balanced Incomplete Block (BIB) design problem [23] has been noted, without realizing that these two are actually the same. In this paper, we introduce a class of Uniform Quorum Systems (UQS), where the quorum size, the size of quorum intersection, and the number of quorums to which each database belongs are all fixed constants. We then show the duality between UQS and the strict BIB, which is, in turn, applied to construct the UQS's used in the proposed scheme.

The performance of the UQS mobility management scheme is evaluated by assigning cost to accessing databases and to call loss due to database failures. We would want the number of databases at the quorum intersection to be large enough to combat the inherent instability of the databases in an ad-hoc network. However, larger size of intersection requires larger size of quorums, which increases the cost of location updates and location queries. The optimal construction of the quorum system that balances this trade-off depends on the stability of each database in the network, as well as the frequency of location updates and queries. We also need to consider the situation, in which so many databases are inaccessible, such that no complete quorum can be found. Such a condition should have very low probability of occurrence.

Following the description of the quorum system mobility management scheme in Section II, we explore the duality between our UQS construction and the BIB design in Section III, introducing four ways of constructing UQS's from families of BIB designs. Section IV presents a framework for evaluating the effectiveness of the UQS in ad-hoc network database failure recovery. The performance of the UQS mobility management schemes is evaluated through cost analysis. Optimal sizes of UQS constructions are also determined under different network traffic and mobility patterns. This leads to the description of a network partitioning scheme in Section V.

II. The Quorum System Mobility Management Scheme over a Virtual Backbone

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*. It dynamically assigns memberships to mobile hosts in the network depending on the communication environment and the network node density. Under ideal conditions, all nodes in the virtual backbone are inter-connected, and every non-backbone node is connected with at least one other backbone node, such that communication between any two nodes is possible. Note that although logically a two-level hierarchy is in place here, the flat network structure is used, such that a connection can be a multi-hop link that spans both the backbone and non-backbone nodes. Thus, routing of the actual traffic is carried by all nodes in the network. In particular, the virtual backbone nodes can communicate between each other through routes that pass through the non-backbone nodes.

The virtual backbone nodes maintain inter-connection among themselves by any appropriate routing methods as described in Section I. As shown later, with the UQS scheme, the optimal number of databases in the virtual backbone is usually small compared with the size of the network. Therefore, for a large ad-hoc network the cost of routing among the virtual backbone nodes is typically very small, compared with routing within the entire network. Besides mobility management, the virtual backbone can also perform other network functions, on which we will not elaborate in this paper, such as channel assignment, flow control, and system security. Thus, the cost of virtual backbone maintenance can by far be offset by the advantages that it brings with it.

The exact algorithm for the initiation and management of the virtual backbone is outside of this paper's scope. During the initial setup, some form of full network routing, such as flooding, is performed once to find the set of nodes that best serve as the virtual backbone(e.g. uniformly distribute within the network). Afterwards, we only need to ensure that when a backbone node has detached from the network, a nearby non-backbone node is recruited to take its place in the virtual backbone. The references [7] and [8] give examples on how backbone sets of nodes in an ad-hoc network can be determined in a distributed fashion without the use of a central controlling entity. Here we emphasize that, in our UQS mobility management scheme, since a connection can be a multi-hop link through both the backbone and non-backbone nodes, the backbone does not have to maintain direct radio contact with every node in the network. Therefore, the algorithm to construct the virtual backbone has much lower complexity than those proposed in [7] and [8].

Figure 1 shows an example of the backbone structure in an ad-hoc network. The squares represent the backbone nodes, and the circles represent the non-backbone nodes.

The location databases are arranged into quorums. In Figure 1 an example quorum system is: $\{\{1, 2, 3\}, \{1, 4, 5\}, \{2, 4, 6\}, \{3, 5, 6\}\}$. When a mobile host, A for example, updates its location, it contacts the closest node that belongs to the backbone set[2], 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[3] that indicates the time of

[2]A mobile host learns about its closest location database through an algorithm such as the intra-zone part of the Zone Routing Protocol proposed in [3].

[3]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

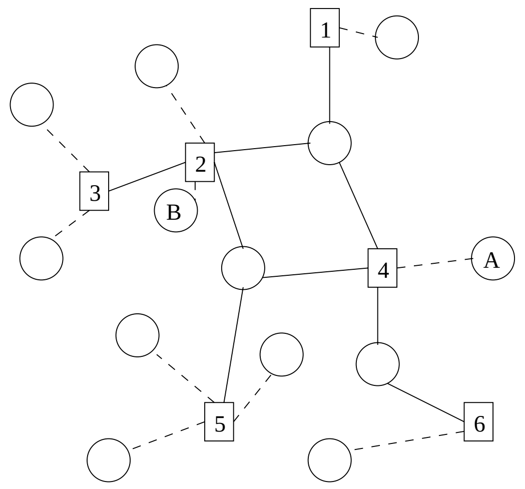

Fig. 1. Virtual Backbone

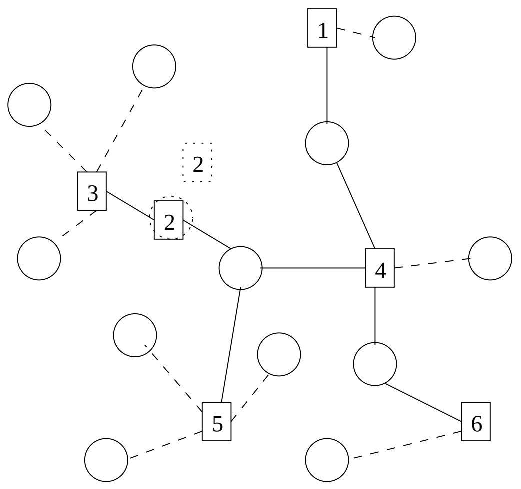

Fig. 2. Virtual Backbone maintenance

update, to one of the quorums, 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 (through multicasting) a quorum, 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. Because of the intersection property of the quorum system, ideally, B can always receive the most current location information of A.

However, in an ad-hoc network, nodes often detach from the rest of the network without warning. The connectivity of the nodes in the backbone set is maintained by a procedure as shown in Figure 2. A backbone node may detach from the network, in which case the topology of the backbone will be rearranged. If a backbone node has been disconnected from the network for more than a threshold amount of time, a new node will be chosen as the replacement. For example, the mobile host B, as shown in Figure 1, may be chosen to replace the detached node containing location database 2. B will set up a new location database with the ID number 2 and become a new member of the backbone set. If the node originally containing location database 2 later reconnects to the network, it will delete the database when it is aware of its long absence and the existence of another database with the same ID. With the replacement, location information stored in the original location database is lost. Therefore we say that a location database *fails* when the node in which it resides detaches from the network.

During the interval from the time at which a location database detaches to the time of its replacement, the database is said to be *inaccessible* to the network nodes. If a virtual backbone mobile host temporarily disconnect from the network and later reconnect back within the time threshold, possibly due to mobile movement or fading and interference in the wireless channel, the database residing in this mobile host is also said to be inaccessible to the network. In this work, we assume that the inaccessible period of a location database is much shorter than the mean time between its failures.

host, and is reset to zero when the maximum value is reached. The maximum value is chosen to be larger than the maximum lifetime of an update message in the network.

We assume that the virtual backbone nodes monitor the accessibility of each other at a frequency much higher than the rate of location updates and queries, so that every backbone node is aware of how many complete quorums are accessible at the time of location update or query. The practicality of this is justified by the relative small size of the virtual backbone, which will be shown in Section IV.

The write quorum and the read quorum intersect to provide access to location information. When there are multiple databases at the intersection of these two quorums, the correct location information will still be available if only some of the databases fail. The UQS construction presented in Section III allows variable number of databases at quorum intersection.

Another way of combating location database failures is to restore the location information in a database after a failure and before the next query. With frequent location database failures, which is often the case in ad-hoc networks, location updating by the mobile hosts can be cost effective, complementing the multiple databases in UQS.

There are three ways in which locations are updated in the databases:

1. Call-origination update: When a mobile host initiates a call, it queries a quorum for the location of the destination and, at the same time, writes its current location into the queried quorum;
2. Location-change update: When a mobile host changes its location (in our example this constitutes changes in its closest database), it updates its new location in a quorum;
3. Periodic update: In order to avoid call loss during long time of lack of activity and immobility, a mobile host periodically sends its location information to the quorum system.

Of the above updating mechanisms, only the frequency of the periodic updates is a parameter in the design of the network. The first two mechanisms, in general, have fixed rates determined by the network traffic and mobility patterns. Call origination can be modeled as a Poisson process. Also, it is reasonable to assume that the time between location-change updates is exponentially distributed due to the irregularities and memoryless nature of the ad-hoc network topology and mobility patterns.

If a call is made based on erroneous location information provided by failed databases or lack of location information, we consider the call as lost. A call is also lost if, upon location query by the calling mobile host, it cannot obtain permission from a complete accessible quorum[4]. Since a database's inaccessible period is assumed to be short, the probability that a complete quorum cannot be found is small compared with the probability that a call is lost due to erroneous information.

The probability of call loss depends on the network stability, the frequency of mobile location updates, and the number of location databases at the intersection of two quorums. Location updates reduce the probability of call loss, but they consume network resources. For every location update, messages need to be sent to all databases in a quorum. Larger quorum size incurs higher cost of location updates. On the other hand, larger quorum size allows larger number of databases at the quorum intersection, thus reducing the probability of call loss. Therefore, the trade-off in the cost due to location updates and the cost due to call loss is closely related to the construction of the quorum system.

III. The Uniform Quorum System Construction

A quorum system has the following parameters:

- n: number of location databases
- q: number of quorums
- k_i: size of the i^{th} quorum, $i = 1, 2, \dots, q$
- m_i: number of quorums that contain the i^{th} database, $i = 1, 2, \dots, n$
- r_{ij}: number of databases shared by two distinct quorums, $i, j = 1, 2, \dots, q$

Any construction of a quorum system can be represented as an incidence matrix N, where N is of size $n \times q$ and

$$N(i,j) = \begin{cases} 1 & \text{if the } i\text{th database is in the } j\text{th quorum,} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

In order to uniformly distribute the load among the databases, it is desirable that each database appears in the same number, m, of quorums. We further require that the number of databases shared by any two distinct quorums be the same, r. In this case, it can be shown that all quorums have the same size, k. We can thus define a Uniform Quorum System (UQS) of the above properties with the parameters (n, q, k, m, r). These parameters are not independent of each other. Moreover, a UQS may not exist for any given set of parameters.

[4]Although it is possible to use partial quorums to store and retrieve location information, in this paper, we assume that a mobile host must access an entire quorum upon location updates and queries. This is justified because of security considerations. See references [13], [14], and [24] for security and availability considerations in quorum systems.

To study the relations between these parameters and the existence conditions of a UQS, we first establish the one-to-one correspondence between the construction of a UQS and the Balanced Incomplete Block (BIB) design problem. A BIB design [23] is an arrangement of v' treatments into b' blocks such that
(i) each block contain k' distinct treatments,
(ii) each treatment appears in r' blocks,
(iii) every pair of treatments appears together in λ' blocks.

Lemma: There is a one-to-one correspondence between a UQS with the parameters (n, q, k, m, r) to a BIB design with the parameters $(v', b', k', r', \lambda')$.

Proof: Given a BIB design with $(v', b', k', r', \lambda')$ and the corresponding incidence matrix N', then

$$n = b', q = v', k = r', m = k', r = \lambda' \tag{2}$$

form a UQS with (n, q, k, m, r), whose incidence matrix is given by

$$N = (N')^T, \tag{3}$$

where $(\cdot)^T$ denotes matrix transposition. Similarly, given a UQS we can alway generate a BIB design whose incidence matrix is the transpose of the incidence matrix of the UQS. *Q.E.D.*

From the properties of BIB designs [23], we immediately have the following corollaries:
1. $nm = qk$,
2. $r(q-1) = k(m-1)$,
3. $q \leq n$,
4. If $k|n$, then $q \leq n - k + 1$.

In optimizing the cost of a location management scheme, as we shall demonstrate in the following section, given some number of location databases in a network, we would like to determine the minimum possible quorum size that provides adequate overlapping between distinct quorums to resist database failures. However, the above are merely necessary conditions for the existence of a UQS. Since no sufficient condition for a BIB design is known in general [23][25], we cannot determine the existence of a UQS given any set of parameters. Moreover, there is no general formula relating n, k, and r. Therefore, we will only study some series of UQS constructions given by readily available BIB designs. Obviously, only those UQS constructions that allow r to vary for a given n are of interest. The following is a list of the series used in our analysis. The original BIB design notations have been translated into our UQS notations.

Series 1 and 2: A necessary and sufficient condition for the existence of a UQS with $m = 3, 4$ and $q \geq m$, due to Hanani's theorem on BIB designs [26], is that

$$r(q-1) \equiv 0 \bmod (m-1) \text{ and } rq(q-1) \equiv 0 \bmod m(m-1) . \tag{4}$$

Series 1 thus consists of UQS with $m = 3$ and all $q \geq 3$ that satisfies Eqn (4). Series 2 consists of UQS with $m = 4$ and all $q \geq 4$ that satisfies Eqn (4). In both cases, $k = r(q-1)/(m-1)$ and $n = rq(q-1)/m(m-1)$.

Series 3: From Sprott's BIB construction [27] using the method of symmetrically repeated differences[23], we have a series of UQS with the parameters r, $m = r$, $k = am$, $q = a(r-1)+1$, and $n = aq$, where a is any positive integer such that q is a prime power.

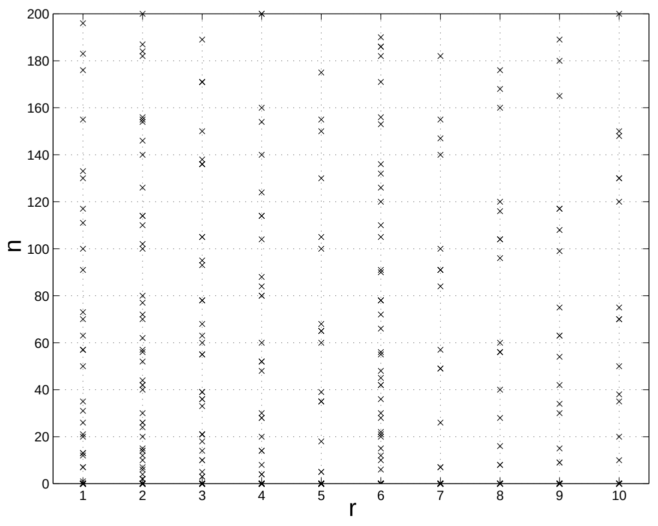

Fig. 3. Possible Number of Databases in the UQS Series

Series 4: Maekawa's finite projective plane BIB design [15], a special case of the projective geometry BIB design [23], gives a UQS construction with $r = 1$, $k = m = s + 1$, and $n = q = s^2 + s + 1$, where s is a prime power. Suppose we have r such UQS's with incidence matrices $N_1, N_2, \ldots, N_r$, we can construct a new UQS by forming the incidence matrix $N = [N_1^T, N_2^T, \ldots, N_r^T]^T$. This series consists of UQS' with the parameters r, $k = r(s+1)$, $m = s+1$, $n = r(s^2+s+1)$, and $q = s^2 + s + 1$.

Series 5: Finally, a special case consists of UQS's with $n = k = r$ and $q = m = 1$. The utilization of these UQS's is equivalent to a scheme with multiple HLRs dynamically generated over a virtual backbone. Obviously they lack the true distribution properties of the other UQS's. However, we include them for completeness in our numerical analysis.

Although all these UQS series allow some flexibility in the choosing of the parameters, by no means all combinations of integer parameters are covered by them. Figure 3 shows the possible values of n ($n \in [1, 200]$) given r for the above Series. In practice, we usually have to pick an allowable n value close to that required by the mobile network system design considerations such as routing and load.

Moreover, even for a fixed r, the correspondence between n and k is not one-to-one, and the relative sizes of n between two quorum systems do not necessarily correspond to the relative sizes of k. In the cost analysis to be presented in the next section, for each r, we tabulate the allowable values of n as shown in Figure 3. When given a prescribed value of UQS size, n_p, we search the table in the ascending order of k, for the first UQS size n_a that is greater than n_p and the one immediately preceding n_a. Among these two possible UQS's, the one whose size is closest to n_p in magnitude is chosen for the numerical analysis.

IV. Optimal UQS Determination through Cost Analysis

In the UQS location management scheme, a mobile updates its location information by writing into a randomly chosen quorum. When a call is initiated, the source mobile queries a randomly chosen quorum for the destination mobile's location. Because of the overlapping property of the UQS, unless the write and the read quorums happen to be the same, there are r databases, in the queried quorum, that contain the current location of the mobile. If all of these r databases have failed within the interval from the last update to the time of the call, the call will be lost, and a penalty to the system results.

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 careful selection of the location update frequency and the construction of the UQS.

To determine the probability that a call to a mobile is lost due to location database failures, we need first to evaluate the distribution of t_r, the time interval between the arrival of a call and the preceding update event, which in turn is a function of the distributions of the call origination, of the updates due to location changes, and of the periodic updates.

Since call-origination and location-change updates are independent, they will be combined into one random process, exponentially distributed with rate $\lambda_u = \lambda_o + \lambda_c$. The time interval, t_u, between a call arrival and the last call-origination or location-change update, is exponentially distributed with the probability density function

$$f_u(t_u) = \lambda_u e^{-\lambda_u t_u} \; . \tag{5}$$

Define t_p as the time interval between a call arrival and the last periodic location update. Then, as shown in the Appendix, t_p has uniform distribution within $[0, T_p]$,

$$f_p(t_p) = \frac{1}{T_p}, \quad 0 \le t_p < T_p \; . \tag{6}$$

Since

$$t_r = \min(t_u, t_p) \; , \tag{7}$$

it has the density function

$$f_r(t_r) = f_u(t_r)[1 - F_p(t_r)] + f_p(t_r)[1 - F_u(t_r)] \; . \tag{8}$$

If the r shared databases between the queried quorum and the quorum that stores the mobile location information fail during this period, the call is lost. Since database failures have Poisson distribution with mean time between failures T_f, the probability that one database fails within t_r is $1 - e^{-\frac{t_r}{T_f}}$. Therefore, without considering the probability of inaccessible databases upon location updates and queries, the expected number of lost calls per unit time would be

$$\begin{aligned} E_{loss} &= \lambda_a \Pr\{r \text{ databases fail within } t_r\} \\ &= \lambda_a \int_0^{T_p} \left(1 - e^{-\frac{t_r}{T_f}}\right)^r f_r(t_r) dt_r \; . \end{aligned} \tag{9}$$

The right hand side of the above equation is slightly larger than the actual value of E_{loss}, since it does not consider the case where the location update and query happen to hit the same quorum, so that all k databases need to have failed to lose a call. However, A UQS that has good distribution properties usually contains many quorums. In this case, the probability that any two randomly chosen quorums are the same is very small. Moreover, a UQS suitable for mobility management should have r large enough to maintain data availability in the presence of database failures. Thus occasionally having k overlapped databases would not substantially improve the call loss probability. Therefore, in the following numerical analysis, we assume that the write and the read quorums always differ.

Taking into account the location update and query failures due to the non-existence of a complete quorum, Equation (9) becomes

$$E_{loss} = \lambda_a(1 - P_Q{}^2) + \lambda_a P_Q{}^2 \int_0^{T_p} \left(1 - e^{-\frac{t_r}{T_f}}\right)^r f_r(t_r)dt_r \ , \tag{10}$$

where P_Q is the steady state probability that a complete quorum can be found.

In a UQS, each database appears in m quorums. Therefore, losing i database, in the worst case, destroys $i \times m$ quorums. This gives the worst-case quorum existence probability

$$\begin{aligned} P_Q &\geq Pr\{i \times m < q\} \\ &= \sum_{i=0}^{\lceil q/m \rceil - 1} \binom{n}{i} p_e{}^i (1 - p_e)^{n-i} \ , \end{aligned} \tag{11}$$

where p_e is the steady state probability of a database being inaccessible. Since the exact degree of quorum system availability(i. e., quorum existence probability) involves complicated factors in addition to the quorum size and intersection properties, and is in general hard to determine [24], we will use this lower bound in the following numerical analysis.

In the general case of routing in ad-hoc networks, the cost of a mobile host accessing a database can be estimated as proportional to 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. Therefore, for any UQS with quorum size k, the expected cost of accessing a randomly chosen quorum is kc_u, independent of the other parameters of the UQS construction.

The sum of the costs due to lost calls and due to updating location databases gives the total cost function:

$$C_{total} = c_l E_{loss} + kc_u \left(\frac{1}{T_p} + \lambda_o + \lambda_c\right) \ . \tag{12}$$

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

The exact value of c_u in a practical ad-hoc network depends on many other parameters, such as the communication environment, the protocols involved, and the degree of link congestion, which can be very diverse for different networks. Thus, in this

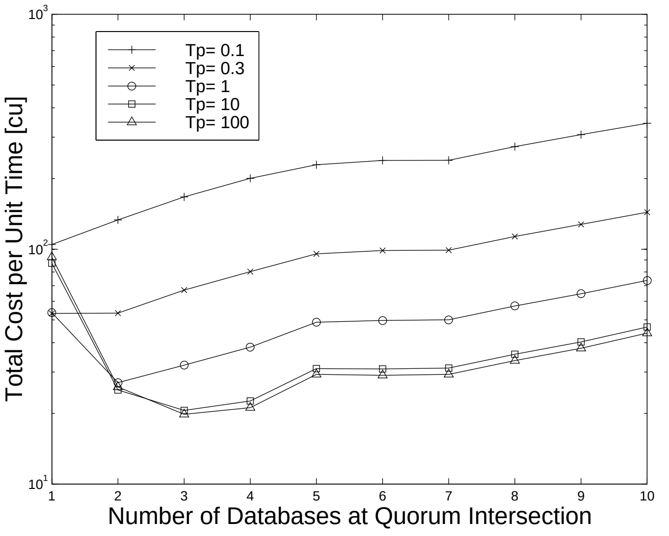

Fig. 4. Cost Optimization at Low Mobility, $\lambda_c = 0.1$, $n \approx 60$, $p_e = 0.005$, and $c_l = 1000$.

paper, we will not dwell on the determination of c_u, but, instead, choose some discrete values from a range of updating cost that covers most practical conditions, and investigate the performance of the UQS scheme under those conditions. In the following numerical analysis, we will normalize c_l and C_{total}, so that they are expressed in units of c_u.

We will also express time units in terms of $1/\lambda_a$. 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 $T_f = 1/\lambda_c$. However, we emphasize that the above framework for cost analysis is applicable to systems without these assumptions as well.

A. The Effect of Mobility on Cost Optimization

Figures 4-7 show the variation of the total cost with the different choices of r for UQS constructions with approximately 60 databases. It is assumed that $p_e = 0.005$, $c_l = 1000$, and $\lambda_c = 0.1$, 1, 10, and 100, respectively. In each figure, T_p is varied from 0.1 to 100. Thus each figure presents a different scenario of mobile host mobility. Consequently, we can determine in each case the minimal cost by joint optimization over the different UQS constructions and different periodic update frequencies. Note that the jitters on the curves are mostly due to the integral nature of the UQS's, as described in Section III. However, even with small variations, the general trends of the curves with respect to r and T_p are still clear.

In Figure 4, $\lambda_c = 0.1$, thus representing the case when the mobile hosts have very low mobility. Since the location databases very infrequently detach from the network in this case, we see that the minimal cost is achieved by $T_p \geq 100$, namely, periodic updates are mostly unnecessary. Also, the minimal cost requires a UQS with $r = 3$.

Figure 5 shows a case of cost optimization at medium mobility, with $\lambda_c = 1$. Here the optimal case is $T_p = 1$ with 4

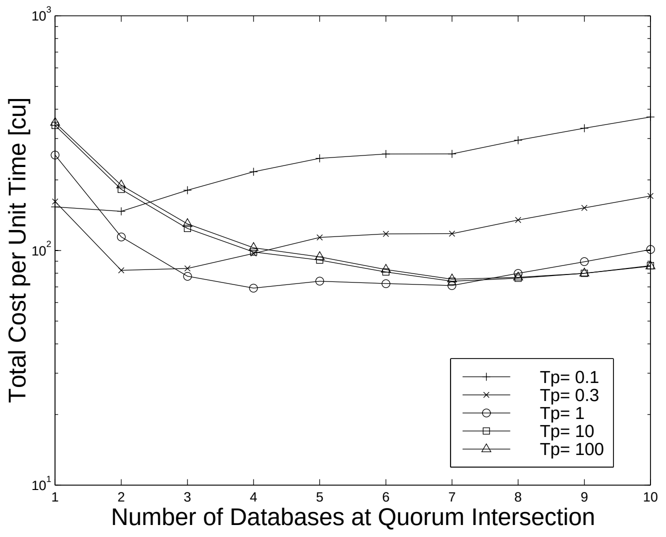

Fig. 5. Cost Optimization at Medium Mobility, $\lambda_c = 1$, $n \approx 60$, $p_e = 0.005$, and $c_l = 1000$.

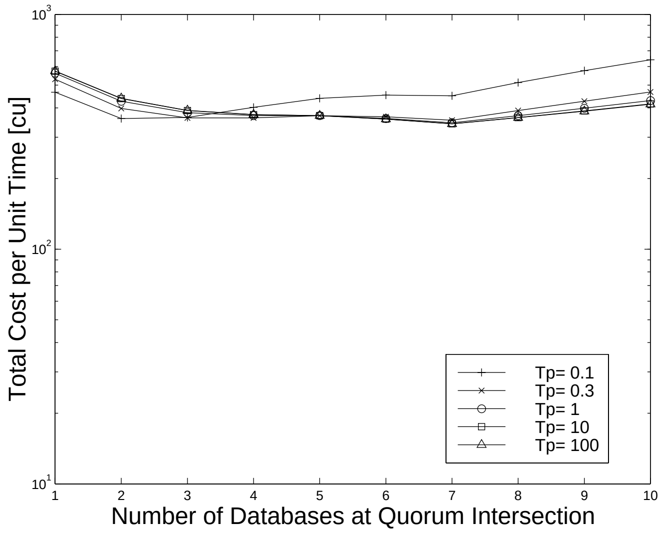

Fig. 6. Cost Optimization at High Mobility, $\lambda_c = 10$, $n \approx 60$, $p_e = 0.005$, and $c_l = 1000$.

databases at quorum intersection. For each curve corresponding to different values of T_p, it is clear that when r is relatively small, thus the quorum size and the location update cost is not significant, increasing r provides higher reliability against location database failures, hence reducing the total cost. As r and the quorum size become larger, however, the cost of location updates dominates, and the total cost increases with r.

Figure 6 shows an interesting case where $\lambda_c = 10$. In this case the different periodic update frequencies all give similar minimal cost. However, the optimal r that achieves the minimal cost is different for each T_p, with smaller r for smaller T_p.

For ad-hoc networks with very fast moving nodes, as shown in Figure 7, where $\lambda_o = 100$, the cost of location updates due to

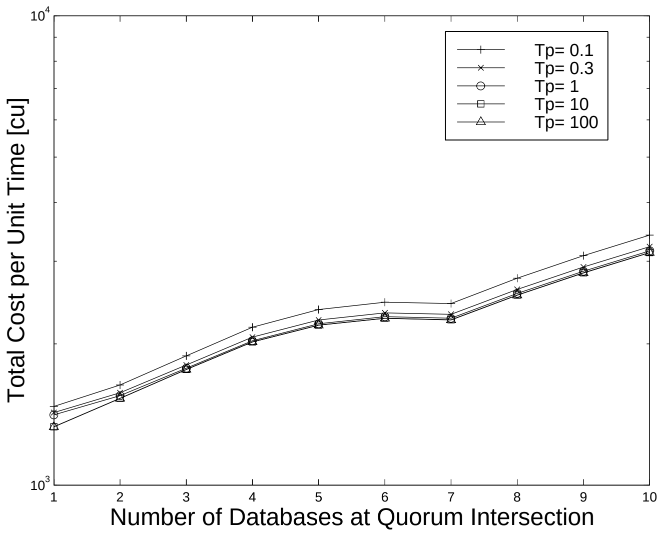

Fig. 7. Cost Optimization at Extremely High Mobility, $\lambda_c = 100$, $n \approx 60$, $p_e = 0.005$, and $c_l = 1000$.

location changes dominates the total cost function. In this case, the effect on database failure recovery due to periodic updates is small, and higher periodic update frequency only increases the total cost, therefore periodic updates should not be used. Since location changes provide very frequent updates, the optimal UQS's should have as small quorum size as possible, i. e., $r = 1$.

B. Joint Optimization over r and T_p

The multiple databases at quorum intersections and the location updates are the two mechanisms that combat network instability in the UQS mobility management scheme. Since r relates to the quorum size, which in turn affects the cost of location updates, r and T_p are not independent of each other in optimizing C_{total}. As the location update frequency decreases, which leads to larger t_r and hence larger E_{loss} as shown in Eqn (10), the optimal r increases to protect the mobile location information. This is demonstrated in the Figures 4-12. For example, in Figure 5, for $T_p = 0.1$ and 0.3, the optimal $r = 2$; for $T_p = 1$, the optimal $r = 4$; and for $T_p \geq 10$, the optimal $r = 7$.

C. The Effect of Availability on Cost Optimization

The availability of a complete quorum upon location updates and queries depends on p_e. Figures 8-10, along with Figure 5, demonstrate the effect of p_e on the optimal T_p and r. All system parameters are the same as those in Figure 5, except that p_e assumes values of: 0, 0.005, 0.01, and 0.1.

Figure 8 represents an ideal case where a detached location database re-attaches or is replaced immediately, thus all databases are always available. The curves in this figure are very similar to those in Figure 5, indicating that a p_e of 0.005 or less has no significant effect on the cost of the system.

The effect of a larger p_e ($p_e = 0.01$) is shown in Figure 9. Comparing this figure with Figure 8, we see that for $r \leq 4$, both cases give similar cost, with slightly higher cost incurred

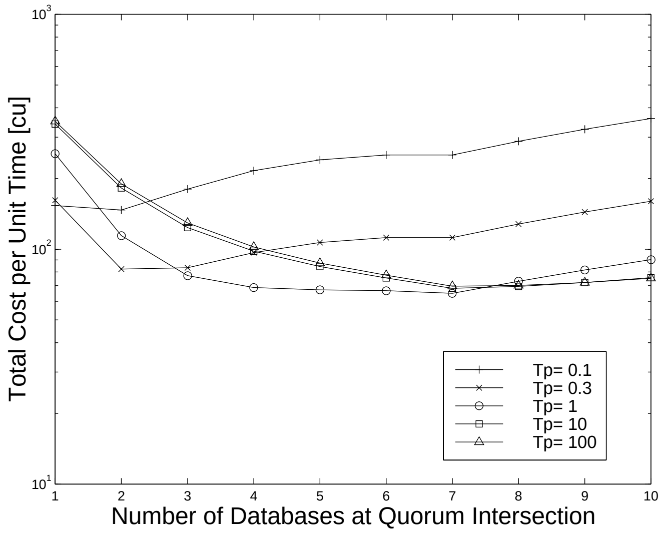

Fig. 8. Cost Optimization with High Database Availability, $p_e = 0$, $\lambda_c = 1$, $n \approx 60$, and $c_l = 1000$.

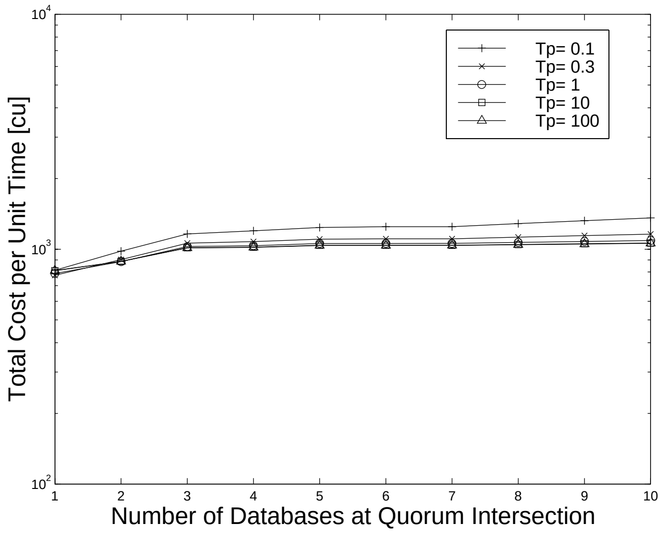

Fig. 10. Cost Optimization with Low Database Availability, $p_e = 0.1$, $\lambda_c = 1$, $n \approx 60$, and $c_l = 1000$.

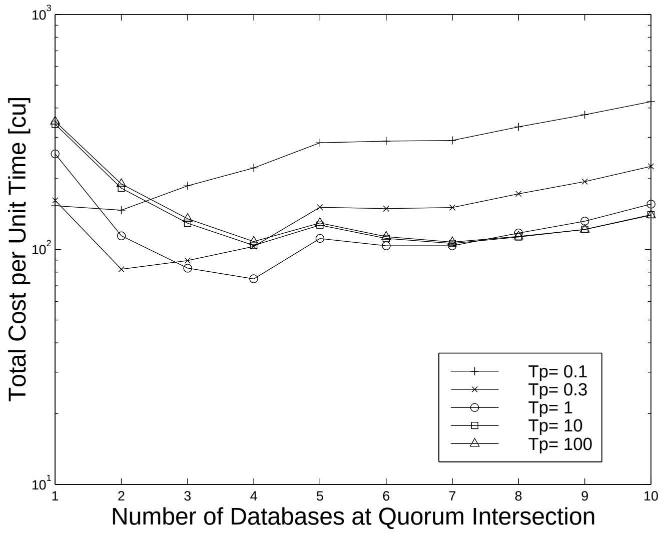

Fig. 9. Cost Optimization with Medium Database Availability, $p_e = 0.01$, $\lambda_c = 1$, $n \approx 60$, and $c_l = 1000$.

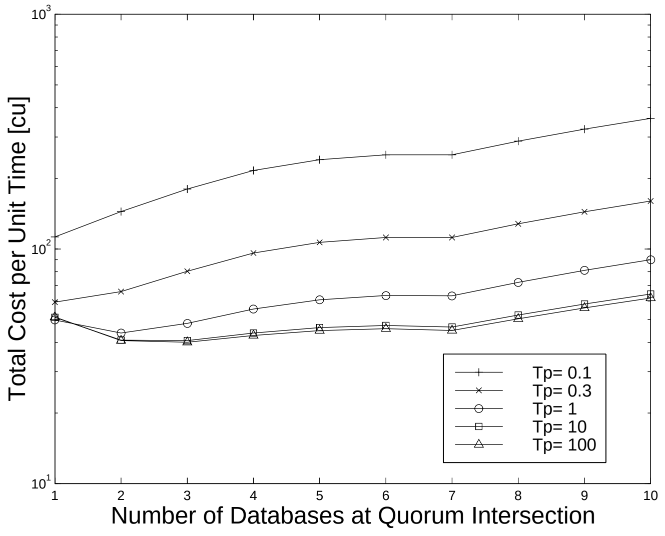

Fig. 11. Cost Optimization with Low Call-Loss Penalty, $c_l = 100$, $p_e = 0$, $n \approx 60$, and $\lambda_c = 1$.

when $p_e = 0.01$. However, when $r \geq 5$, the performance of the system with higher p_e severely degrades, indicating that UQS's with larger r have lower worst-case quorum availability. The minimal cost is achieved for the same values of T_p but a slightly smaller r.

Figure 10 shows a case where the databases are inaccessible 10% of the time. In this case, the UQS scheme performs poorly with all T_p and r.

D. The Effect of Call-Loss Penalty on Cost Optimization

Figures 11 and 12, along with Figure 8, demonstrate the effect of the cost due to lost calls on the optimization of T_p and r. All the system parameters are the same in these figures, except that c_l assumes values of: 100, 1000, and 10000. These figures suggest that call-loss cost is the weight that shifts the joint optimization over T_p and r. With low call-loss cost, the system is optimal when the periodic update frequency is low and the quorum intersection is small, as shown in Figure 11. With higher call-loss cost, the optimal point shifts towards higher update frequencies and larger r, as shown in Figure 12.

E. The Selection of UQS Size and the Cost Optimization

From the UQS constructions in Section III, the quorum size k is not fixed for a given r. Since k is proportional to the location update cost, C_{total} generally decreases as k decreases, provided that p_e is small enough so that the availability of quorums does

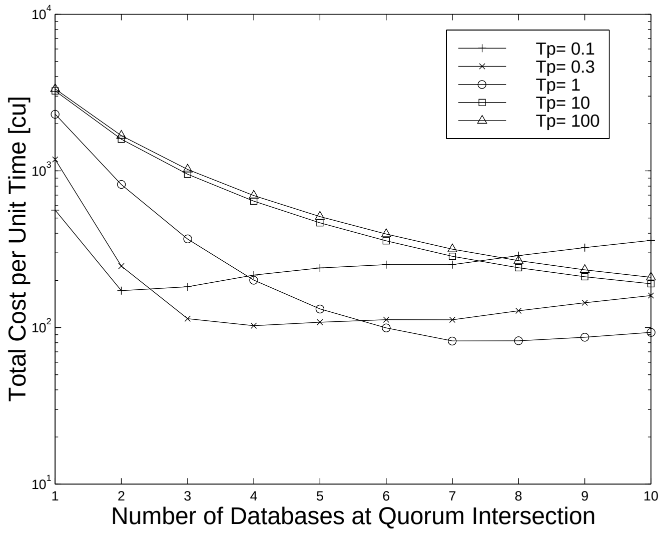

Fig. 12. Cost Optimization with High Call-Loss Penalty, $q = 10000$, $p_e = 0$, $n \approx 60$, and $\lambda_c = 1$.

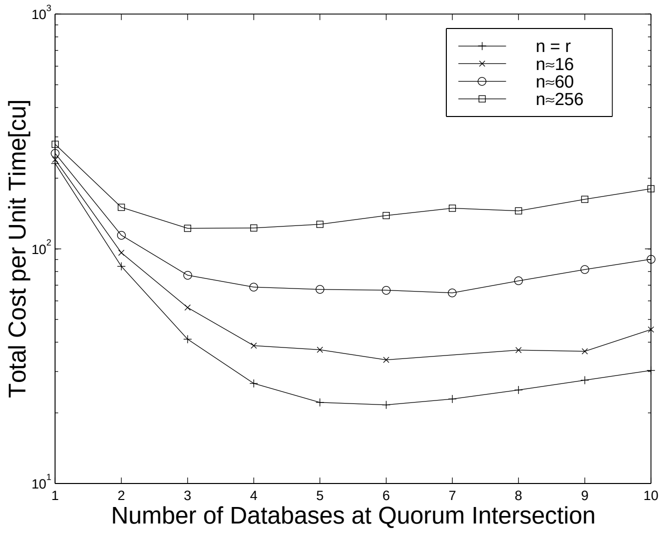

Fig. 13. Optimization over n, $T_f = 1$, $T_p = 1$, $\lambda_o = 1$, $c_l = 1000$, and $p_e = 0$

not significantly affect the total cost. Since k also reflects the total number of databases in a UQS, smaller UQS's generally are more desirable in reducing the total cost. Figure 13 verifies this with plots of total cost versus r for different UQS sizes n. The system parameters are $T_f = 1$, $T_p = 1$, $\lambda_o = 1$, $c_l = 1000$, and $p_e = 0$. The sizes of the UQS's are, respectively, $n = r$, $n \approx 16$, $n \approx 60$, and $n \approx 256$. Here we see that the minimum is achieved by the smallest UQS that has the intersection of $r = 5$. With close inspection, we see that this is the degenerative case where $n = k = r = 5$ and $m = q = 1$, which is also the "adaptive HLR" case described in Section I.

As the inaccessible probability of the location databases changes, so does the optimal n. Figure 14 shows plots of the cost

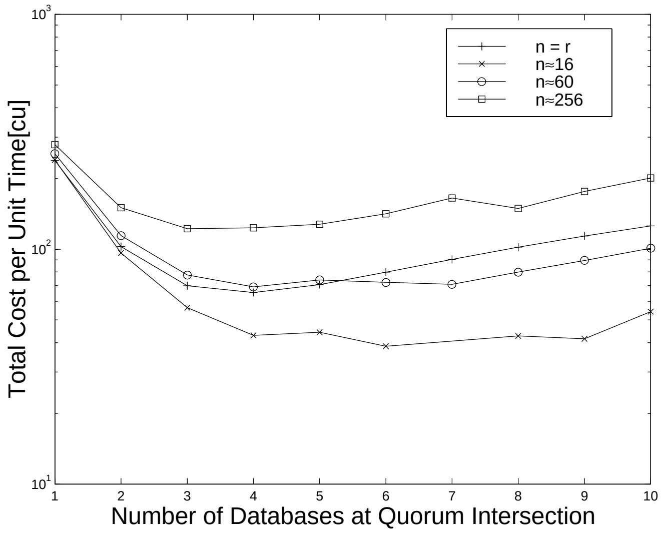

Fig. 14. Optimization over n, $T_f = 1$, $T_p = 1$, $\lambda_o = 1$, $c_l = 1000$, and $p_e = 0.005$

optimization over r and n in a network with the same parameters as those in Figure 13, except that p_e is raised to 0.005. The plots suggest that the UQS's with approximately 16 databases give the lowest cost. Thus, in the case where some nodes are inaccessible for even a very short time, which is extremely common in ad-hoc networks, the general UQS scheme outperforms the "adaptive HLR" scheme.

The optimal number of location database, based on the above analysis of the cost of mobility management, is generally quite small. However, for many reasons, a network may require more than this optimal number of location databases. We present in the next section a scheme that partitions the network nodes to maintain the efficiency in mobility management.

V. Ad-Hoc Network Partitioning

In practice, the number of location databases in an ad-hoc network is usually determined by conditions other than the UQS, such as the mobile host loading limitations or security requirements. As shown in Section IV, the optimal UQS size, n^*, is usually small. For the case that the actual number of databases is larger than the optimal UQS size, we propose to partition the databases into groups of approximate size n^*, each forming the optimal UQS, and each responsible for the location management of a logical partition of the network nodes.

The partitioning of the network nodes can be accomplished by calculating modulo p of the mobile ID numbers, where p is the number of partitions. Each partition manages an n^*-node backbone, forming the optimal UQS with the n^* location databases, which store only location information of the mobile hosts within the partition. All p backbones are inter-connected, i. e., every node in one backbone has a route to every node in another backbone, so that location update and query messages can be sent across the entire network.

Figure 15 illustrates an ad-hoc network, which requires 12 location databases, partitioned into two parts, each with a back-

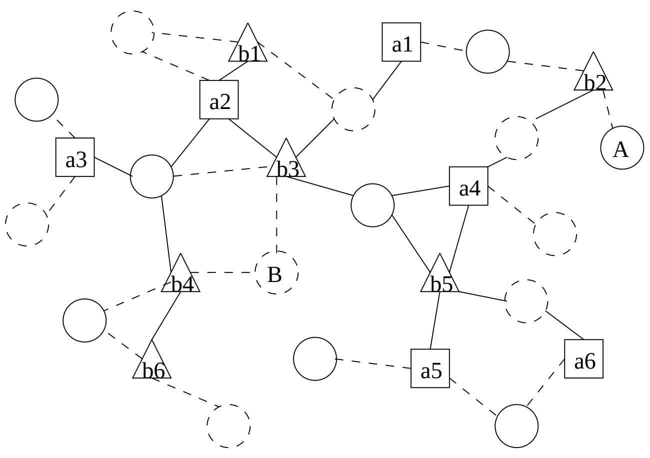

Fig. 15. Network Partition

bone of 6 nodes. The solid circles represent non-backbone nodes that belong to the same partition as the square backbone nodes, and the dotted circles represent non-backbone nodes that belong to the same partition as the triangular backbone nodes. In this case, since $b2$ is the nearest location database from A, the location of node A is defined as $b2$, although A and $b2$ are not in the same partition. When A performs an update, it contacts $b2$, which recognizes that A belongs to the "square" partition and relays the location information of A to a quorum, say, $\{a1, a4, a5\}$. When B initiates a call to A, it contacts the nearest location database $b4$, which randomly queries a "square" quorum, say, $\{a1, a2, a3\}$. Since $a1$ has the current location of A, namely the ID of $b2$, it sends this information back to $b4$ and hence B.

With negligible overhead in accessing inter-partition quorums, this scheme allows the selection of the optimal UQS that achieves minimal cost of location updates and cost due to lost calls. For example, assuming traffic and mobility pattern parameters as those in Figure 14, suppose that the location server loading requirements in an ad-hoc network demand 64 location databases. When all of the databases are used to form a quorum system, the minimal total cost for mobility management is approximately 80 [c_u/mobile/unit time]. If we partition the network nodes into four groups with 16 location databases, a minimal total cost of 30 [c_u/mobile/unit time] can be achieved.

However, partitioning compromises some distributive features of quorum systems, such as independent numbering and system security. Therefore, the truly optimal size of partitioning of the system may depend on other considerations, rather than the mobility management cost alone.

VI. Conclusions

Duplication and distribution of mobile host location information is the key to mobility management in ad-hoc networks, whose connectivity is inherently unstable due to the lack of pre-existing fixed infrastructures. In this paper, we introduce the Uniform Quorum System and propose a scheme, in which mobile location databases form a virtual backbone arranged into a UQS, and where copies of the location information of a mobile can be stored and retrieved in a random and distributed fashion. Since the virtual backbone is dynamically maintained, adaptive to network traffic and mobility patterns, and an updating or querying mobile host randomly chooses any accessible quorum of location databases, the system is robust against database failures, which is characteristically frequent in an ad-hoc network.

The duality between UQS and BIB has been demonstrated here. Constructions of UQS's are proposed through families of BIB designs. A framework is presented for evaluation of the cost of location management using UQS's. We show how the quorum size and the number of databases at quorum intersection affect the trade-off between the cost of duplicated location updates and the penalty due to call loss. We also investigate how the network traffic and mobility patterns, location database stability, and the relative cost of call loss and location update, affect the optimization of the quorum constructions and the mobile periodic location update frequencies.

The optimal virtual backbone size based on mobility management cost is usually small compared with the network size. Therefore maintaining the virtual backbone incurs relatively small overhead in terms of network bandwidth and computation resources. What we gain is a quorum system storing the network node location information, which, comparing with simply multicasting every node's location to every other node, provides higher degree of stability and security in mobility management. This is especially so in the Reconfigurable Wireless Networks [3], where the network span is large and the topology change is frequent.

We have shown that the "adaptive HLR" scheme based on a dynamic virtual backbone is a limiting case of the UQS scheme. The numerical results indicate that, in the practical ad-hoc network environment where nodes swing in and out of the network, the general UQS scheme outperforms the "adaptive HLR" scheme.

The numerical analysis suggests that, in the UQS scheme, the mobile hosts in an ad-hoc network should employ periodic location updating when the network node mobility is comparable to the call arrival rate. The periodic update frequency, in this case, should also be comparable to the call arrival rate.

Among UQS's with similar number of databases, the ones with more overlapped databases should be used for systems with higher location database failure rate, less frequent location updates, or heavier call-loss penalties, and vice versa. Within the intervals of parameters under consideration in the analysis, the optimal r typically ranges from 2 to 6, with $r = 1$ or $r \geq 7$ only under extreme conditions.

Short periods of databases being inaccessible, e. g., $p_e < 0.01$ in our analysis, do not significantly alter the joint optimization of periodic location updates and quorum intersection size. However, as the probability that a database is inaccessible becomes higher, the UQS performance degrades, especially for those with large quorum intersection, i. e., $r \geq 5$.

We also show that for the same number of databases at quorum intersection, a smaller UQS gives lower total cost, provided that such UQS has relatively low probability of system-wide failure due to the absence of a complete quorum. Since a network may have larger than the optimal number of databases, we propose a scheme which partitions the network nodes into groups where the databases can form the optimal quorum system. This ensures the efficiency of the UQS scheme in terms of mobility management, while allowing the virtual backbone perform other network functions.

Future work on the UQS scheme includes computing a tighter lower bound of quorum availability and investigating other forms of virtual backbone arrangement that are suitable for mobility management. The practical aspects of the quorum system mobility management in the multiple service provider environment, such as resource sharing and provider independent bill generation, also need to be addressed.

Appendix

The following is the derivation of the probability density function of the time interval, T, between a call arrival and the immediately preceding periodic update.

Call arrivals are assumed to be Poisson with rate λ_a. Let N be the total number of call arrivals within an inter-periodic-update interval, of length T_p, during which a call arrives. Suppose this is the k^{th} call arrival since the last periodic update. Let T_k be the time interval between this call arrival and the last periodic update. Further, define $N(t)$ as the number of call arrivals within a time interval of length t. Given $N = n$, T_k has the conditional probability distribution

$$\begin{aligned}
& Pr\{T_k \le t \mid N = n\} \\
= & \sum_{l=k}^{n} Pr\{N(t) = l \mid N(T_p) = n\} \\
= & \sum_{l=k}^{n} \frac{Pr\{N(t) = l, N(T_p) - N(t) = n - l\}}{Pr\{N(T_p) = n\}} \\
= & \sum_{l=k}^{n} \frac{Pr\{N(t) = l, N(T_p - t) = n - l\}}{Pr\{N(T_p) = n\}} \\
= & \sum_{l=k}^{n} \frac{e^{-\lambda_a t}\frac{(\lambda_a t)^l}{l!} \cdot e^{-\lambda_a (T_p - t)} \frac{[\lambda_a (T_p - t)]^{n-l}}{(n-l)!}}{e^{-\lambda_a T_p}\frac{(\lambda_a T_p)^l}{n!}} \\
= & \sum_{l=k}^{n} \left(\begin{array}{c} n \\ l \end{array} \right) \frac{t^l (T_p - t)^{n-l}}{{T_p}^n} . \qquad (13)
\end{aligned}$$

The probability that a call arrival is the k^{th} one is $1/N$, given that there are N call arrivals within the inter-periodic-update interval. Therefore, the conditional density function of T, given $N = n$, is

$$\begin{aligned}
& f_T(t \mid N = n) \\
= & \sum_{k=1}^{n} \frac{1}{n} \cdot \frac{d}{dt} Pr\{T_k \le t \mid N = n\} \\
= & \frac{d}{dt} \sum_{k=1}^{n} \sum_{l=k}^{n} \frac{1}{n} \left(\begin{array}{c} n \\ l \end{array} \right) \frac{t^l (T_p - t)^{n-l}}{{T_p}^n} \\
= & \frac{1}{{T_p}^n} \frac{d}{dt} \sum_{l=1}^{n} \sum_{k=1}^{l} \frac{1}{n} \left(\begin{array}{c} n \\ l \end{array} \right) t^l (T_p - t)^{n-l} \\
= & \frac{1}{{T_p}^n} \frac{d}{dt} \sum_{l=1}^{n} \frac{l}{n} \left(\begin{array}{c} n \\ l \end{array} \right) t^l (T_p - t)^{n-l} \\
= & \frac{1}{{T_p}^n} \frac{d}{dt} \sum_{l=1}^{n} \left(\begin{array}{c} n-1 \\ l-1 \end{array} \right) t^l (T_p - t)^{n-l} \qquad (14)
\end{aligned}$$

Substituting $m = l - 1$, we have

$$\begin{aligned}
& f_T(t \mid N = n) \\
= & \frac{1}{{T_p}^n} \frac{d}{dt} \sum_{m=0}^{n-1} \left(\begin{array}{c} n-1 \\ m \end{array} \right) t^m (T_p - t)^{n-m-1} \cdot t \\
= & \frac{1}{{T_p}^n} \frac{d}{dt} (t + T_p - t)^{n-1} \cdot t \\
= & \frac{1}{{T_p}^n} \frac{d}{dt} {T_p}^{n-1} t \\
= & \frac{1}{T_p} . \qquad (15)
\end{aligned}$$

Finally, the density function of T is given by

$$\begin{aligned}
f_T(t) &= \sum_{n=1}^{\infty} f_T(t \mid N = n) Pr\{N = n \mid N \ge 1\} \\
&= \frac{1}{T_p} \sum_{n=1}^{\infty} Pr\{N = n \mid N \ge 1\} \\
&= \frac{1}{T_p} . \qquad (16)
\end{aligned}$$

References

[1] S. Mohan and R. Jain, “Two User Location Strategies for Personal Communications Services,” *IEEE Personal Communications*, First Quarter, 1994.

[2] Z. J. Haas and Y-B. Lin, “On Optimizing the Location Update Costs in the Presence of Database Failures,” *ACM/Baltzer Wireless Networks Journal*, vol 4, no 5, pp. 419-426, 1998.

[3] Z. J. Haas and Marc R. Pearlman, “Providing Ad-Hoc Connectivity with the Reconfigurable Wireless Networks,” *Proceedings of the ACM SIGCOMM’98*, September, 1998.

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

[5] 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.

[6] 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.

[7] 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.

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

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

[10] R. Prakash and M. Singhal, “A Dynamic Approach to Location Management in Mobile Computing Systems,” *Proceedings of the 8^{th} International Conference on Software Engineering and Knowledge Engineering (SEKE’96)*, pp. 488-495, June, 1996.

[11] R. Prakash, Z. J. Haas, and M. Singhal, “Distributed and Load-Balanced Location Management for Mobile Systems,” submitted to *IEEE/ACM Transactions on Networking*.

[12] R. Prakash and M. Singhal, “Dynamic Hashing + Quorum = Efficient Location Management for Mobile Computing Systems,” *Proceedings of ACM Symposium on Principles of Distributed Computing*, August, 1997.
[13] D. Agrawal and A. El Abbadi, “Quorum Consensus Algorithms for Secure and Reliable Data,” *Proceedings of the Seventh Symposium on Reliable Distributed Systems*, pp. 44-53, October 1988.
[14] M. Naor and A. Wool, “Access Control and Signatures via Quorum Secret Sharing,” *3rd ACM Conf. on Comp. and Comm.*, pp. 157-168, March 1996.
[15] M. Maekawa, “A $\sqrt{N}$ Algorithm for Mutual Exclusion in Decentralized Systems,” *ACM Trans. on Computer Systems*, Vol.3, No.2, pp. 145-159, May 1985.
[16] S. Y. Cheung, M. H. Ammar, and M. Ahamad, “The Grid Protocol: A High Performance Scheme for Maintaining Replicated Data,” *Proc. 6th IEEE Int. Conf. Data Engineering*, pp. 438-445, 1990.
[17] D. Agrawal and A. El Abbadi, “An Efficient and Fault-Tolerant Solution for Distributed Mutual Exclusion,” *ACM Trans. Comp. Sys.*, 9(1), pp. 1-20, 1991.
[18] D. Peleg and A. Wool, “Crumbling Walls: A Class of Practical and Efficient Quorum Systems,” *Proceedings of the 14th ACM Symposium on Principle of Distributed Computing*, pp. 120-129, 1995.
[19] W. K. Ng and C. V. Ravishankar, “Coterie Templates: A New Quorum Construction Method”, *Proc. 15th IEEE Int. Conf. Distributed Computing Systems*, pp. 92-99, 1995.
[20] B. Awerbuch and D. Peleg, “Concurrent Online Tracking of Mobile Users,” *ACM SIGCOMM’91* in *Computer Communication Review*, vol.21, no.4, pp. 221-233, 1991.
[21] S. J. Mullender and P. M. B. Vitanyi, “Distributed Match-Making,” *Algorithmica*, vol.3, pp. 367-391, 1988.
[22] A. Nakajima, “Sizes of a Symmetric Coterie,” *Transactions of Information Processing Society of Japan*, vol.36, no.6, pp. 1467-1473, 1995.
[23] A. Dey, *Theory of Block Designs*, 1986.
[24] D. Peleg and A. Wool, “The Availability of Quorum Systems,” *Information and Computation*, 123, pp. 210-223, 1995.
[25] H. J. Ryser, *Combinatorial Mathematics*, 1963.
[26] H. Hanani, “Balanced Incomplete Block Designs and Related Designs,” *Discrete Mathematics*, 11, pp. 255-369, 1975.
[27] D. A. Sprott, “Some Series of Balanced Incomplete Block Designs,” *Sankhya*, 17, pp. 185-192, 1956.