Probabilistic Quorum Systems in Wireless Ad Hoc Networks

ROY FRIEDMAN
Technion – Israel Institute of Technology, Haifa
GABRIEL KLIOT
Microsoft Research
and
CHEN AVIN
Ben Gurion University of The Negev

Quorums are a basic construct in solving many fundamental distributed computing problems. One of the known ways of making quorums scalable and efficient is by weakening their intersection guarantee to being probabilistic. This article explores several access strategies for implementing probabilistic quorums in ad hoc networks. In particular, we present the first detailed study of asymmetric probabilistic biquorum systems, that allow to mix different access strategies and different quorums sizes, while guaranteeing the desired intersection probability. We show the advantages of asymmetric probabilistic biquorum systems in ad hoc networks. Such an asymmetric construction is also useful for other types of networks with nonuniform access costs (e.g, peer-to-peer networks). The article includes a formal analysis of these approaches backed up by an extensive simulation-based study. The study explores the impact of various parameters such as network size, network density, mobility speed, and churn. In particular, we show that one of the strategies that uses random walks exhibits the smallest communication overhead, thus being very attractive for ad hoc networks.

Categories and Subject Descriptors: C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*Distributed application*

General Terms: Algorithms, Design

The work was primarily done while G. Kliot was a Ph.D. student in the Computer Science Department in the Technion.
Based on a shorter preliminary version of this article which appears in *Proceedings of the IEEE 38th International Conference on Dependable Systems and Networks (DSN'08)*. © 2008 IEEE.
Authors' addresses: R. Friedman, Computer Science Department, Technion – Israel Institute of Technology, Haifa 32000, Israel; email: roy@cs.technion.ac.il; G. Kliot, Microsoft Research, One Microsoft Way, Redmond, WA 98052; email: Gabriel.Kliot@microsoft.com; C. Avin, Communication Systems Engineering Department, Ben Gurion University of The Negev, Beer Sheva, Israel; email: avin@cse.bgu.ac.il
Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or direct commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or permissions@acm.org.
© 2010 ACM 0734-2071/2010/09-ART7 $10.00
DOI 10.1145/1841313.1841315 http://doi.acm.org/10.1145/1841313.1841315

Additional Key Words and Phrases: Distributed middleware, location service, quorums systems, random walks, wireless ad hoc networks

ACM Reference Format:

Friedman, R., Kliot, G., and Avin, C. 2010. Probabilistic quorum systems in wireless ad hoc networks. ACM Trans. Comput. Syst. 28, 3, Article 7 (September 2010), 50 pages.
DOI = 10.1145/1841313.1841315 http://doi.acm.org/10.1145/1841313.1841315

1. INTRODUCTION

Quorums are a basic construction in many distributed systems. They can be used as building blocks (or at least as a design pattern) in solving various fundamental problems such as Consensus [Guerraoui and Raynal 2004], distributed dictionaries and location services [Friedman and Kliot 2006], distributed storage [Attiya et al. 1995; Lynch and Shvartsman 2002], etc. While most implementations of quorum systems are deterministic, some works have suggested the use of probabilistic quorums as a way of improving their resilience, efficiency, and scalability [Malkhi et al. 2001].

The idea behind quorums is that they ensure intersection between subsets of nodes (we formally define quorums and biquorums in Section 2.1). The intersection property enables maintaining consistency of actions taken by nodes of a distributed system. This is achieved by contacting a quorum of nodes before taking an action that could change the state of the system, or before completing operations that might conflict other operations. This ensures that any two such actions are seen by at least one common node, which enables the detection and elimination of conflicts.

In ad hoc networks [Toh 2002], location services are one of the most important services for many of the envisioned applications, as they enable users to find information and services stored by others. As discussed in Friedman and Kliot [2006], a very large percentage of location services are based on biquorums whether they are built explicitly or implicitly as part of the overall solution. Recently, there have been several attempts to solve Consensus and distributed storage in ad hoc networks, such as Chockler et al. [2005], Dolev et al. [2003], and Luo et al. [2003]. As mentioned before, these also require quorums.

In this article we investigate the implementation of scalable probabilistic quorums in ad hoc networks. Ad hoc networks are unique, compared to wired networks, in several aspects. In wireless ad hoc networks, routing and flooding are extremely expensive. Hence, applications and services developed for these networks should avoid multiple hop routing and flooding as much as possible, and instead aspire to rely solely on local one-hop message exchanges. The dynamic nature of ad hoc networks (caused by churn and node mobility) makes the usage of strict deterministic quorums highly costly. Therefore, for the sake of scale and efficiency, we relax the requirements of the quorum system to probabilistic ones, similar to the seminal work of Malkhi et al. [2001].

One could potentially use geographical knowledge for construction of quorum systems in ad hoc networks (e.g., Li et al. [2000] and Tchakarov and Vaidya [2004]). However, as GPS and other accurate positioning techniques may not always be available, and since the network's boundaries are not always known, in this article we look for quorum systems that do not rely on geographical knowledge.

Contributions of this Work. A central contribution of our work is the orderly introduction and study of several schemes for accessing probabilistic quorums in ad hoc networks. We study the performance of the proposed schemes both analytically and by simulations. The simulations explore the behavior of the various schemes under changing parameters such as network scale, network density, mobility, and churn (failures/departures and joins). We also explore several optimizations and heuristics to deal with these issues, and demonstrate their impact through simulations.

One of the schemes we investigate is based on Random Walks (RW). The use of random walks in wireless ad hoc networks has been previously proposed to solve various problems such as membership construction [Dolev et al. 2002; Bar-Yossef et al. 2008], reliable multicast [Dolev et al. 2002], routing [Servetto and Barrenechea 2002], and querying [Avin and Brito 2004]. Random walks are attractive for ad hoc networks since they require neither multihop routing nor broadcasting, which are expensive in ad hoc networks [Bar-Yossef et al. 2008]. Moreover, they offer fine-grain control over the communication overhead as well as early halting capabilities. In addition, random walks are highly robust to failures and mobility with very little overhead. These benefits also show up in our simulations, as elaborated later in this article.

Another important contribution of this work is the introduction of an *asymmetric* probabilistic biquorum system. Specifically, in previous works, for example, Luo et al. [2004, 2003] and Malkhi et al. [2001] accesses to all quorums of a probabilistic biquorum system are performed using the same access strategy.[1] In that respect, previously known probabilistic biquorum systems are *symmetric*. In particular, we prove a “mix-and-match” lemma. This lemma shows that it is possible to combine a random access strategy with any other access strategy (and each may have a different quorum size) and still obtain intersection with a desired probability. This is an important enabling result for implementing probabilistic biquorum systems in any network in which the access cost is nonuniform, such as ad hoc networks and peer-to-peer networks [Gkantsidis et al. 2004; Lv et al. 2002]. This is because in a network with nonuniform communication cost, accessing a random set of nodes means having to contact at least some far-away (or “expensive”) nodes on each quorum access. Our result shows that in fact only one type of quorum access (e.g., advertise or update) needs to be “expensive,” while the other (e.g., lookup or read) can be optimized to only access the closest (or “cheapest”) nodes. Specifically, our study demonstrates that such asymmetric biquorum systems indeed offer superior performance compared to symmetric ones in ad hoc networks.

Additionally, we provide a number of useful formal results about random walks in *random geometric graphs* [Penrose 2003], which are a natural model

[1]In Luo et al. [2004, 2003], the authors use the term “asymmetric probabilistic quorums,” however, the asymmetry in Luo et al. [2004, 2003] refers to different quorum sizes and not different access strategies.

of wireless networks. In the context of random walks, the cost of accessing a quorum using a random-walk-based strategy is measured by the *Partial Cover Time (PCT(i))*, that is, the expected time it takes the random walks to visit a set of i distinct nodes. We provide a novel result about the partial cover time of random walks in random geometric graphs and show that covering a sublinear number of nodes i takes a number of random walk steps that is linear in i, and therefore this access strategy is indeed "cheap." In addition, we provide a definition of the *crossing time* of random walks and a lower bound on the crossing time in random geometric graphs. This translates to the expected time it will take two random walks to cross (i.e., meet) each other. These results help us reason about the costs and benefits of the various access strategies, as well as biquorum systems as a whole.

The theoretical parts of this work assume a unit disk graph model with perfect equal transmission ranges. In particular, the formal analysis of random walks is based on the random geometric graph model [Penrose 2003]. All our results are validated through an extensive simulation study, performed on the JIST/SWANS simulator from Cornell,[2] which models mobility, collisions, and signal propagation, and implements the entire 802.11 MAC. Both the formal model and simulations model, including the mobility pattern, signal reception model, and other simulations parameters, are described and explained in Sections 2.3 and 2.4.

Article Road Map. For methodological reasons, we first present the basic concepts of probabilistic biquorum systems and the various access strategies, and only then turn to the mix-and-match lemma and the simulations-based evaluation. Specifically, we start by introducing some preliminaries and system model in Section 2. Section 3 describes the metrics used to evaluate quorum systems. In Section 4 we present a number of general strategies for accessing a single quorum. We then show in Section 5 how to mix these strategies and implement several different probabilistic ϵ-intersecting quorum systems and discuss their properties analytically. Quorum maintenance in the face of churn and mobility is discussed in Section 6 and certain optimizations of the basic access strategies are discussed in Section 7. Section 8 presents the simulation study. We discuss related work in Section 9 and conclude with a discussion in Section 10.

2. PRELIMINARIES

This section introduces the main concepts and definitions used throughout this work. To help the reader follow and relate to these concepts, we include a summary of the main acronyms in Figure 1.

2.1 Quorums and Biquorums

Intuitively, a quorum system is a set of subsets such that every two subsets intersect. Moreover, a biquorum system consists of two sets of subsets such

[2]`http://jist.ece.cornell.edu/`

Q	Quorum
|Q|	Quorum size
RANDOM	Types of quorum access strategies
PATH	
UNIQUE-PATH	
FLOODING	
1-ε	Intersection probability
TTL	Time to live
RW	Random walk
PCT	Partial cover time of the random walk
$G^2(n,r)$	Random geometric graph
n	Number of nodes
d_{avg}	Average node degree

Fig. 1. Acronyms.

that each subset in one set intersects with each subset in the other set. Shortly, we provide a formal definition of these notions, following the works of Garcia-Molina and Barbara [1985], Gramoli [2007], Herlihy [1986], and Mitchell et al. [1992].

Definition 2.1 (*Set System*). A set system $\mathcal{S}$ over a universe U is a set of subsets of U.

Definition 2.2 (*Quorum System*). A quorum system $\mathcal{Q}$ over a universe U is a set system over U such that for any $Q_1, Q_2 \in \mathcal{Q}$, $Q_1 \cap Q_2 \neq \emptyset$.

Definition 2.3 (*Bi-quorum System*). A biquorum system $\mathcal{Q}$ over a universe U is a couple of set systems $(\mathcal{Q}_1, \mathcal{Q}_2)$ such that for any $Q_1 \in \mathcal{Q}_1$ and $Q_2 \in \mathcal{Q}_2$, $Q_1 \cap Q_2 \neq \emptyset$.

In this work we focus on biquorums. We will also refer to them here as `lookup` and `advertise` quorums given that biquorums are often used in conjunction with lookup and advertise operations.[3] However, the discussion applies the same for any biquorum system.

A data discovery service as well as any distributed dictionary can be implemented using an advertise/lookup quorum system as follows: Publishing a data item is implemented by contacting all members of a single `advertise` quorum and having them store the information. Looking up the data is performed by contacting a `lookup` quorum. The intersection between any `advertise` quorum and any `lookup` quorum ensures that if a data item has been published, it will be found by the lookup operation.

[3]We discuss implementing read/write registers via quorums in Section 10.

2.2 Probabilistic Quorums

In *probabilistic quorums* [Malkhi et al. 2001], a quorum system is not fixed a priori, but is rather picked in a probabilistic manner for each interaction with the quorum system. For example, in the case of biquorums, such as lookup/advertise quorums, it is ensured that each `lookup` quorum intersects with every `advertise` quorum with a given probability.

2.3 Ad Hoc Network System Model

Consider a wireless network comprising a finite set of nodes V, located arbitrarily in the plane. A node in the system is a device owning an omnidirectional antenna that enables wireless communication. Let X_i for $i = 1, 2, \ldots, |V|$ denote the location of the nodes. For simplicity, we also use X_i to refer to the ith node itself. We denote by $|X_i - X_j|$ the Euclidian distance between node X_i and node X_j.

Communication Model. We consider two possible models for a successful reception of a transmission over one hop, which are commonly used in the literature [Gupta and Kumar 2000; Keshavarz-Haddad et al. 2006].

—*The Protocol Model*. In this model, each node X_i is associated with a specific transmission range r_i. A transmission from X_i is received successfully by X_j if and only if $|X_i - X_j| \leq r_i$ and for every other simultaneously transmitting node X_k, $|X_k - X_j| \geq (1+\triangle)r_k$. $\triangle > 0$ is the *interference parameter*. To simplify the analysis, we assume that all transmission ranges are the same: for each $X_i \in V, r_i = r$.

—*The Physical Model*. Let $\mathcal{T} \subseteq V$ be the set of nodes simultaneously transmitting at some time instant, and denote by P_k the transmit power of node $X_k \in \mathcal{T}$. The transmission from node X_i is received successfully by X_j if and only if

$$\frac{\frac{P_i}{|X_i - X_j|^{\alpha}}}{N_0 + \sum_{X_k \in \mathcal{T}, X_k \neq X_i} \frac{P_k}{|X_k - X_j|^{\alpha}}} \geq \beta.$$

In this model, a minimum Signal-to-Interference-plus-Noise Ratio (SINR) of β is necessary for successful reception. The ambient noise power level is denoted N_0 and the signal decays deterministically with distance r as $1/r^{\alpha}$. As typical to this model, we assume that $\alpha = 2$. In our simulations, we use homogeneous transmit powers: for each $X_i \in V$, $P_i = P$.

Theoretical Graph Model. We carry all our analysis in the protocol communication model, which greatly simplifies the formal analysis without affecting the access strategies. In addition, all our results are validated by a simulation study performed in the more realistic physical model, which includes a realistic signal propagation model, signal interference, distortions, background noise, unidirectional links, etc., as elaborated in Section 2.4.

In the protocol communication model, all nodes that are located within the transmission range r of a node X_i form the set of X_i's potential neighbors. The combination of the nodes and the neighborhood relations forms a wireless ad hoc network. The resulting network connectivity graph $G = (V, E)$ is a 2-dimensional *unit disk* graph, in which n nodes are embedded in the surface of a unit torus,[4] and any two nodes within Euclidean distance r of each other are connected. When the nodes are placed uniformly at random on the surface the resulting graph is known as a *Random Geometric Graph* (RGG) [Penrose 2003] and is denoted by $\mathcal{G}^2(n, r)$. Specifically, the $\mathcal{G}^2(n, r)$ graph is often used to model the network connectivity graph of 2-dimensional wireless ad hoc networks and sensor networks [Gupta and Kumar 1998, 2000].

We assume that nodes do not know their position and we do not use any geographic knowledge. Yet, each node knows all of its temporal direct potential neighbors (nodes with which it can currently communicate directly). This can be implemented, for example, by a simple heartbeat mechanism that is present in any case in most routing protocols in ad hoc networks [IETF Mobile Ad-Hoc Networks Working Group 2008]. In addition, a node can communicate with its nondirect neighbors (if they exist), if other nodes agree to relay its messages.

New nodes may join and existing nodes may leave the network at any time, either gracefully or by suffering a crash failure. Nodes that crash or leave the network may rejoin it later. The rate at which nodes join and leave the system in known as the *churn rate* of the system. We assume that the network remains continuously connected.

2.4 Simulations Setup

Our simulations were performed using the JiST/SWANS simulator [Barr et al. 2005] from Cornell University, which is a discrete event-based network simulator that includes an accurate model of a full networking stack. All simulation parameters are summarized in Figure 2.

The signal interference model used was RadioNoiseAdditive, which is based on a cumulative noise computation with SINR and capture effect. This is equivalent to the physical model of successful reception and is also identical to the interference model used in Glomosim and NS2 version 2.33 simulators (refer to Kliot [2008] for more details). The ideal reception range, without any collisions, was set to 200m and the average number of neighbors, d_{avg}, ranged from 7 to 25 in various simulations (default $d_{\text{avg}} = 10$). This was achieved by scaling the area size according to $a^2 = \frac{\pi r^2 n}{d_{\text{avg}}}$ and resulted in all networks being connected (according to the connectivity result of Gupta and Kumar [1998], d_{avg} should be $\pi r^2 n = C \ln n$, for $C > 1$ and in our case $d_{\text{avg}} = 7$ bounded $C \ln n$ for all n's we used). The nodes were placed at uniformly random locations in a square plane. All simulations were performed on networks of 50, 100, 200, 400, and 800 nodes.

We use AODV for multihop routing when accessing quorums selected by the RANDOM access strategy. The mobility pattern was the Random Waypoint model with different speeds. The default speed of movement ranged from

[4]In practice, the network is flat rather than a torus. However, this does not affect the access strategies, but greatly simplifies the formal model. At any event, our simulations are carried on a flat topology with a physical reception model.

PHY	
Signal Propagation model (PathLoss)	Two-Ray ground reflection
Signal Interference model	Cumulative noise with SNR and capture effect
Transmit power	15 dBm (31.62 mW)
Receive Thresh (RXThresh in ns2)	–71 dBm (7.9432e-8 mW)
Sensitivity Thresh (CSThresh in ns2)	–77 dBm (1.9952e-8 mW)
Background noise (thermal noise)	–101 dBm (8.0080e-11 mW)
SNR (β) (CPThresh in ns2)	10
TX/RX Antenna gain	0 dBm (1 mW)
Fading	None
Ideal Reception range	200 meter
Carrier Sensing Range	299 meter
MAC	
Modulation	DSSS with long preamble and PLCP header
Slot Time	20 μs
DIFS	50 μs
Bandwidth	11 Mbps for unicast, 2 Mbps for broadcast
Simulation Scenarios	
Message size	512 bytes + IP + MAC + PHY headers
Node count	50, 100, 200, 400, 800
Density (# of one hop neighbors)	**10 default.** Varying: 7, 10, 15, 20, 25.
Mobility	Random WayPoint, **default speed 0.5-2 m/s** Varying max speed: 2,5,10,20 m/s, pause 30s
Routing protocol (when applicable)	AODV
Heartbeat cycle	10 sec
# of advertisements	100
# of lookups	1000 (by 25 random nodes)
Java pseudo random number generator, initialized with the current time in millis as seed	

Fig. 2. Simulation parameters.

0.5–2 m/s, which corresponds to slow and fast walking speeds, and an average pause time of 30 seconds.

2.5 Relaxed Intersection Requirements

As mentioned earlier, quorum systems typically serve as a building block for higher-level services. Many such services deal with data storage and retrieval, be it a data location service [Friedman and Kliot 2006], a dictionary, bulletin board, shared objects and registers [Gramoli 2007; Lynch 1996], etc. The requirement that each `lookup` quorum intersects with *each* `advertise` quorum is important to ensure that each retrieval of data can return the most recent value that was stored in the service, as shown, for example, in Attiya et al. [1995].

However, certain services may settle for a weaker semantics in which data retrieval operations are allowed to return any previously stored value rather than the last one. Similarly, the usage pattern of some services may be such that each data item is only written once. In particular, the latter is known as immutable data and is popular in many large storage systems (e.g., in the storage system described by Rowstron and Druschel [2001]). For such services, the intersection requirement can be relaxed so that each `lookup` quorum needs only to intersect with *some (but not necessarily all)* previously accessed `advertise` quorum. This relaxed requirement may enable certain optimizations, as discussed in Sections 6.1 and 7.1.

3. QUORUM SYSTEMS METRICS

Any implementation of a probabilistic quorum system can be analyzed according to the following quality measures [Malkhi et al. 2001].

Intersection probability. A probabilistic quorum system $\mathcal{Q}$ is ε-intersecting if the probability of any pair of quorums to intersect is at least $1 - \varepsilon$.
Formally, Let $\mathcal{Q}$ be a set system, let w be an access strategy for $\mathcal{Q}$, and let $0 < \varepsilon < 1$ be given. The tuple $\langle \mathcal{Q}, \varepsilon \rangle$ is an ε-intersecting quorum system if $Pr[Q \cap Q' \neq \emptyset] \geq 1 - \varepsilon$, where the probability is taken with respect to the strategy w.

Access cost. The cost (in messages) of accessing a quorum.

Load. The load of requests on a single node. The target is to balance the load equally among the nodes.

Failure resilience. The resilience of the quorum system to failures. It is measured by two parameters:
(1) *Fault tolerance* of a quorum system $\mathcal{Q}$ is the size of the smallest set of nodes that intersects all quorums in $\mathcal{Q}$ (i.e., the minimal number of nodes whose crash will leave the system without any quorum). As shown by Malkhi et al. [2001], the fault tolerance of a probabilistic quorum system of size $k\sqrt{n}$ is $n - k\sqrt{n} + 1 = \Omega(n)$. In an ad hoc network it is also required that all network nodes form a connected graph (see Section 6 for an elaborative discussion on connectivity of ad hoc networks).
(2) *Failure probability* of a quorum system is the probability that the system becomes disabled when individual nodes crash independently with a fixed probability p. As shown by Malkhi et al. [2001], the failure probability of a probabilistic quorum system of size $k\sqrt{n}$ is $e^{-\Omega(n)}$ for all $p \leq 1 - \frac{k}{\sqrt{n}}$. Again, in ad hoc networks, the network graph must remain connected.
Failure resilience refers to the resilience of the whole quorum system to failures, rather than the resilience of a single quorum. As long as the entire quorum system has not failed, a new live quorum can be found. But it does not say anything about the liveness or a chance of survival of a previously accessed quorum. Therefore, failure resilience is not enough to measure the resilience of a dynamic quorum system. For this reason we introduce the following measure.

Degradation rate. The rate of change in the intersection probability, as a function of network churn. For probabilistic quorums, this metric translates

	RANDOM		PATH	FLOODING
	Membership based	Sampling based		
Accessed Nodes	Random Uniform	Random Uniform	Arbitrary	Arbitrary
General Network	$\lvert Q\rvert \, Diameter$	$\lvert Q\rvert T_{mix}$	$PCT(\lvert Q\rvert)$	$\lvert Q\rvert$
Random Geometric Graph	$\lvert Q\rvert\sqrt{\frac{n}{\ln(n)}}$	$\lvert Q\rvert n$	$\lvert Q\rvert$, for $\lvert Q\rvert = o(n)$	$\lvert Q\rvert$
Need Routing	Yes	No	No	No
Need Membership Service	Yes	No	No	No
# Lookup Replies	Multiple*	Multiple*	One	Multiple
Early Lookup Halting**	No*	No*	Yes	No

* unless accessed serially ** valid only for write -once data

Fig. 3. Asymptotic and qualitative comparison of different access strategies, for general networks and for Random Geometric Graphs. $|Q|$ stands for quorum size, *Diameter* for the diameter of the network, PCT for partial cover time of the random walk, T_{mix} for mixing time of the random walk.

to the probability that two quorums accessed at different times will intersect despite the fact that between these two accesses some nodes have crashed or new nodes have joined. Hence, degradation rate captures the resilience of a single quorum in the face of dynamic changes. The degradation rate helps to determine when the quorum system should be reconfigured, or refreshed, in order to recover from failures, node departures, and joins.

4. QUORUM ACCESS STRATEGIES

An *access strategy* defines the way in which a client trying to access a probabilistic quorum propagates its requests. The access strategy may impact all the measures of a quorum system we presented before. In this work we focus on three main strategies: RANDOM, PATH, and FLOODING. RANDOM simply accesses a set of random, uniformly chosen nodes. PATH is a random walk which traverses the network graph until it covers a sufficient set of different nodes. FLOODING performs a limited-scope flooding of the network, which covers a set of different nodes.

In addition, we consider a number of significant optimizations which can make some of the strategies even more appealing. The main novelty of our approach is an ability to mix these strategies in different ways when implementing probabilistic biquorum systems, achieving various trade-offs discussed shortly. More specifically, we show that in order to construct probabilistic quorum systems, some quorums can be accessed by optimized, nonrandom strategies. This is in contrast with the previous methods [Luo et al. 2003; Malkhi et al. 2001] which used a RANDOM strategy to access every quorum.

Figure 3 provides a summary of the asymptotic costs and qualitative properties of various access strategies for general networks and for random geometric graphs. We elaborate on each of the strategies and the corresponding entries in the table of Figure 3. The first row refers to the type of accessed nodes: random uniform or arbitrary. The second and third rows capture the cost (the number of messages) to access $|Q|$ nodes. Number of lookup replies refers to whether multiple redundant replies will be sent in response for a lookup access and early halting refers to the ability to stop the lookup operation the moment the looked-up object is found, without the need to access the full quorum. The early halting is valid only under the relaxed intersection requirement of Section 2.5 and its implementation is described as a location service-specific optimization in Section 7.1. We now turn to the details.

4.1 RANDOM

In this method, a quorum (be it `lookup` or `advertise`) is simply any random, uniformly chosen set of nodes Q. We consider two implementations of this access strategy.

Membership service-based implementation. If full membership knowledge is available (it can be obtained through a standard membership service [Chockler et al. 2001], implemented, e.g., by every node occasionally flooding the network with its id), a node can simply randomly select node ids from its membership list. Alternatively, nodes can utilize a random membership service for ad hoc networks, such as RaWMS [Bar-Yossef et al. 2008], which provides every node with a set of uniform-randomly chosen node ids.

Once nodes' ids for a given quorum have been fixed, accessing this quorum can be done by sending a message to each of these nodes through unicast routing. For a quorum of size $|Q|$ we need to send messages to $|Q|$ nodes. Thus, at the application level, the cost is $|Q|$. However, since we are operating over ad hoc networks, the true number has to take into account the cost of multihop routing, which includes both the cost of using the routes and establishing the routes. In the general case, the expected cost of using the route is $\Theta(|Q| \cdot \mathit{Diameter})$, while *Diameter* denotes the diameter of the network.

Cost in random geometric graphs. It is well known [Gupta and Kumar 1998] that the diameter of a random geometric graph with transmission range r is $\Theta(1/r)$ and the minimal r to guarantee network connectivity is $\Omega(\sqrt{\frac{\ln n}{n}})$. Thus, assuming the nodes are uniformly distributed, the price of accessing a quorum of this type is

$$\Theta(|Q| \cdot \mathit{Diameter}) = \Theta(|Q| \cdot 1/r) = O\left(|Q|\sqrt{\frac{n}{\ln n}}\right).$$

When $|Q| = \Theta(\sqrt{n})$, the cost is $O(\frac{n}{\sqrt{\ln n}})$.

As for the cost of establishing the routes, it is hard to predict analytically. This cost also depends on route reuse. In slow-moving ad hoc networks with low churn rate, it is best to reuse the same quorum between consecutive invocations as long as all its members are reachable. This amortizes the initial route discovery cost over several requests.

Direct sampling-based implementation. If no membership service exists, a quorum can be picked directly by using a sampling service. One possible implementation of a random uniform sampling service for ad hoc networks based on random walks is described in Bar-Yossef et al. [2008]. Generally, random walks sample nodes in a nonuniform manner, proportionally to nodes' degrees. To provide uniform samples, the sampling algorithm in Bar-Yossef et al. [2008] utilizes a special form of random walk, called a Maximum Degree RW (MD RW). Every uniform sample is obtained by a single MD RW, whose length equals the network mixing time.

Using this method, the RANDOM quorum can be accessed directly by starting an appropriate number of Maximum Degree RWs every time a quorum should be accessed. The data item is then published (or looked for) at the end node. Since two or more random walks may end in the same node, then for a quorum of size $|Q|$, we may need more than $|Q|$ random walks. However, for similar arguments as in the birthday paradox, as long as $|Q| \leq O(\sqrt{n})$, the chance of such collisions is very small. Hence, no more than $|Q|$ random walks are needed (for a more precise analysis refer to Bar-Yossef et al. [2008]). As a result, the cost of accessing a RANDOM quorum by this method is $\Theta(|Q| \cdot T_{mix})$, while T_{mix} denotes the network graph mixing time.

Cost in random geometric graphs. The mixing time of a MD RW in random geometric graphs was studied in Bar-Yossef et al. [2008] and was found to be approximately $n/2$. Thus, the total communication complexity of accessing a quorum of size $\Theta(\sqrt{n})$ in this way is $\Theta(n\sqrt{n})$. Yet, here we never invoke routing.

Notice that in the presence of failures, mobility, or churn, simply attempting to access a uniform random sample of $|Q|$ nodes may result in fewer than $|Q|$ replies. This is unless care is taken to ensure reliability. Overcoming such problems is discussed in Section 6.

4.2 PATH

Another way to pick a quorum is by performing a single *simple random walk* which traverses the underlying network graph until it visits $|Q|$ different nodes. Naturally, the size of the quorum, $|Q|$, should be set such as to guarantee the intersection property of the quorum system. At every step, a simple random walk picks one of the neighbors of a current node uniformly at random and moves to this neighbor. In addition, the random walk counts the number of distinct nodes it has visited. This can be implemented, for example, by storing the list of all visited nodes in the random walk header. Another way to implement it is to attach a unique identifier (e.g., a tuple comprising the starting node id and a sequence number) to each random walk, and have each node store the identifier of each random walk that passes through it; the random walk increments its counter only if it is new to the node. Since our goal is to use short random walks (in the order of $\sqrt{n}$), the list of visited nodes in the header does not introduce a significant additional communication overhead. This recording also helps in sending replies on the reverse path of the walk. In order to do the same when nodes record the ids of random walks, each node must also record the last prior hop from which the random walk has reached this node.

Random walk advantages. The biggest advantage of a random walk is that it does not require multiple-hop routing. In addition, the random walk implementation does not assume anything about the properties of the underlaying network graph (aside from being connected). A random walk simply proceeds until a required number of distinct nodes has been encountered. In addition, whenever it is enough to satisfy the relaxed intersection requirement of Section 2.5, PATH quorums have an ability to terminate earlier before visiting all $|Q|$ nodes. We describe this optimization of the PATH access strategy in Section 7.1, together with other location service-specific optimizations.

In order to estimate the efficiency of the random walk-based method, we must estimate the average number of steps that takes a random walk to visit i different nodes. Generally a random walk can revisit the same nodes more than once and may even include loops.

The number of steps required for a simple random walk to visit i different nodes is called the *Partial Cover Time* and is denoted by $PCT_G(i)$ (for a given graph G). Formally, for $v \in V$, let $PCT_v(i)$ be the expected number of steps needed for a simple random walk starting at v to visit i distinct nodes in G, and the partial cover time of G is $PCT_G(i) = \max_v PCT_v(i)$. The *cover time*, C_G, of G is the expected time to visit all nodes in G, that is, $C_G = \max_v PCT_G(n)$.

PCT of random geometric graphs. It is already known that for random geometric graphs $\mathcal{G}^2(n,r)$ and $0 \le c < 1$, $PCT_G(cn) \le O(n)$ [Avin and Brito 2004] and the full cover time ($c = 1$) is $PCT_G(n) = O(n \log n)$ [Avin and Ercal 2007]. We now extend the result of Avin and Brito [2004] to achieve a tighter bound on covering a sublinear number of nodes.

The following theorem establishes a novel result about partial cover time of random geometric graphs: covering $t = o(n)$ different nodes in $\mathcal{G}^2(n,r)$ is linear in t.

THEOREM 4.1. *Let $t(n) = o(n)$. For a constant $c > 1$, if $r^2 \ge \frac{c \cdot 8 \log n}{n}$, then, for large enough n, with high probability for $\mathcal{G}^2(n,r)$*

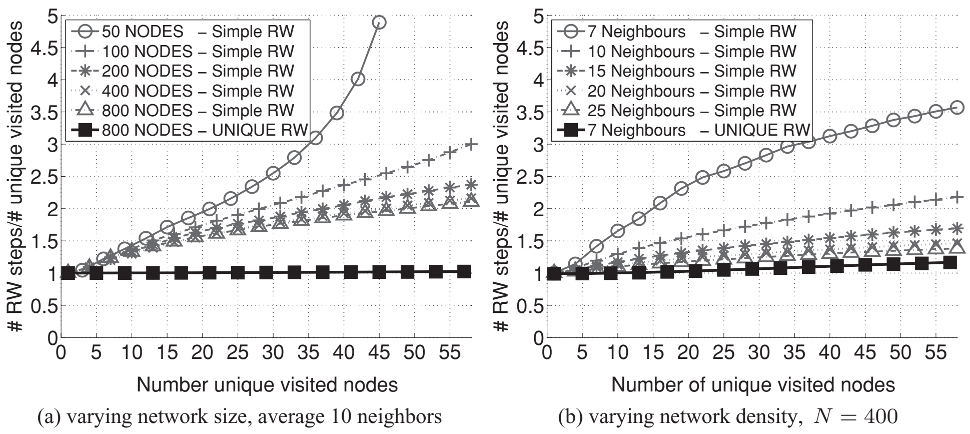
$$PCT_{\mathcal{G}}(t) \le 2\alpha t,$$

where α is a constant.

PROOF IDEA. The detailed proof is deferred to the Appendix. It is based on bounding the expected number of visits to each node during the walk of length $t = o(n)$ by a constant α not dependent on n and then showing that the number of distinct nodes visited by the random walk of this length is at least $t/(2\alpha)$.

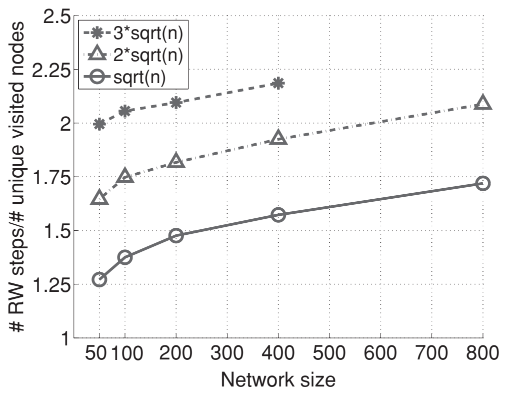
Remark. Theorem 4.1 says that the time to visit t distinct nodes is bounded by $2\alpha t$. For example, for $t \approx \sqrt{n}$ it is $2\alpha t = O(\sqrt{n})$, that is, it is linear in t. The preceding holds for a fixed value of t (which is a function of n) and for all large enough values of n. Note, however, that the value of α may be different for different t (e.g., α for $t = \sqrt{n}$ is different from the α for $t = n^{\frac{3}{4}}$) and therefore we evaluate it empirically.

Empirical study of the partial cover time. Since Theorem 4.1 establishes an asymptotic result without specifying the exact constants, we have performed



(a) varying network size, average 10 neighbors

(b) varying network density, $N = 400$



(c) varying the network size and quorum size, average 10 neighbors, simple RW

Fig. 4. RW Partial Cover Time: the evolution of the number of random walk steps required to visit unique nodes. UNIQUE-RW almost never revisits nodes (for a small number of visited nodes).

an empirical study of the partial cover time. Figure 4 depicts simulation results for the number of different nodes visited by a single random walk (and UNIQUE-PATH RW explained shortly), for different network sizes, densities, and quorum sizes. In these runs, performed on the same setting as reported in Section 2.4, we run a random walk until it visits a fixed number of different nodes and count the number of required steps (averaging across multiple random walks and runs). In the graphs, the y-axis represents the number of random walk steps divided by the number of unique nodes visited by the random walk. Hence, the graphs show the cost paid by the random walk in order to reach a certain number of unique nodes.

According to Figures 4(a) and 4(c), for example, with $n = 800$, in order to visit $\sqrt{800} = 28$ nodes, a simple random walk has $PCT_{\mathcal{G}}(\sqrt{n}) = 1.7\sqrt{n}$ (the simple random walk length is $1.7 \times 28 \sim 48$). For smaller networks, the same constant of 1.7 is enough: $PCT_{\mathcal{G}}(\sqrt{n}) \leq 1.7\sqrt{n}$ for any $n \leq 800$, as can be easily seen in Figure 4(c). Actually, $PCT_{\mathcal{G}}(i)$ for even larger values of i is still linear in i. For example, for $n = 100$, $PCT_{\mathcal{G}}(50) = 2.6 \times 50 = 127$, thus $PCT_{\mathcal{G}}(n/2) \approx 1.3n$.

Figure 4(b) depicts the influence of the network density on $PCT_{\mathcal{G}}$. Intuitively, in sparse networks a random walk is expected to revisit the same nodes more often than in denser ones. This is because when a node has only a few neighbors, the random walk has limited choices at every step. On the other hand, a dense network, in which every node has a very large number of neighbors, may resemble a complete graph. A complete graph is known to have a very small PCT (for example, $PCT_{complete}(n/2) = \ln(2)n \approx 0.69n$ on a complete graph, which can be shown by a simple balls and bins argument [Motwani and Raghavan 1995]). Note that the sparsest network that is still connected has 7 neighbors on average. Less than 7 neighbors resulted in disconnected networks with multiple partitions in almost all our simulations, as also predicted by Gupta and Kumar [1998]. Even in this network, $PCT_{\mathcal{G}}(\sqrt{n}) = 2.5\sqrt{n}$, for $n = 400$.

4.3 UNIQUE-PATH

An optimization of the PATH strategy is to perform the random walk in a way that avoids visiting the same nodes more than once, also known as a *self-avoiding* random walk [Madras and Slade 1993]. That is, at every step, a random walk picks at random one of the neighbors of a current node that has not been visited yet and moves to this neighbor. In a rare event that all the neighbors of a current node have been visited by the random walk, an arbitrary random neighbor is chosen (as in a simple random walk).

A self-avoiding random walk is expected to have a shorter partial cover time than a simple random walk over the same network, since more different nodes are visited by the same number of steps. We have empirically explored the covering properties of a self-avoiding random walk. As can be seen from Figure 4, empirically, indeed UNIQUE-PATH almost never revisits a node (at least for $|Q| = O(\sqrt{n})$). As opposed to the simple random walk, the self-avoiding random walk is almost unaffected by the network density. Even at the sparsest network of 7 neighbors, $PCT_{\mathcal{G}}(60) = 70$, for $n = 400$. This comprises the biggest advantage of the UNIQUE-PATH strategy.

In addition, note that the UNIQUE-PATH strategy incurs the same communication bit-complexity as the simple PATH. This is because we need to remember the ids of the visited nodes in the random walk message header anyway, to count the number of different visited nodes in order to make sure the correct number of quorum nodes have been accessed.

If the quorum system is to be used for a location service (or any other service that requires quorum nodes to reply to the originating node) the node that stores the location information has to send back a reply specifying the actual location. If PATH or UNIQUE-PATH are used, it would be more beneficial to send the reply back on the reverse path of the random walk instead of invoking a costly routing to send the reply. Additional random walk-related optimizations and implementation details are described in Sections 6 and 7.

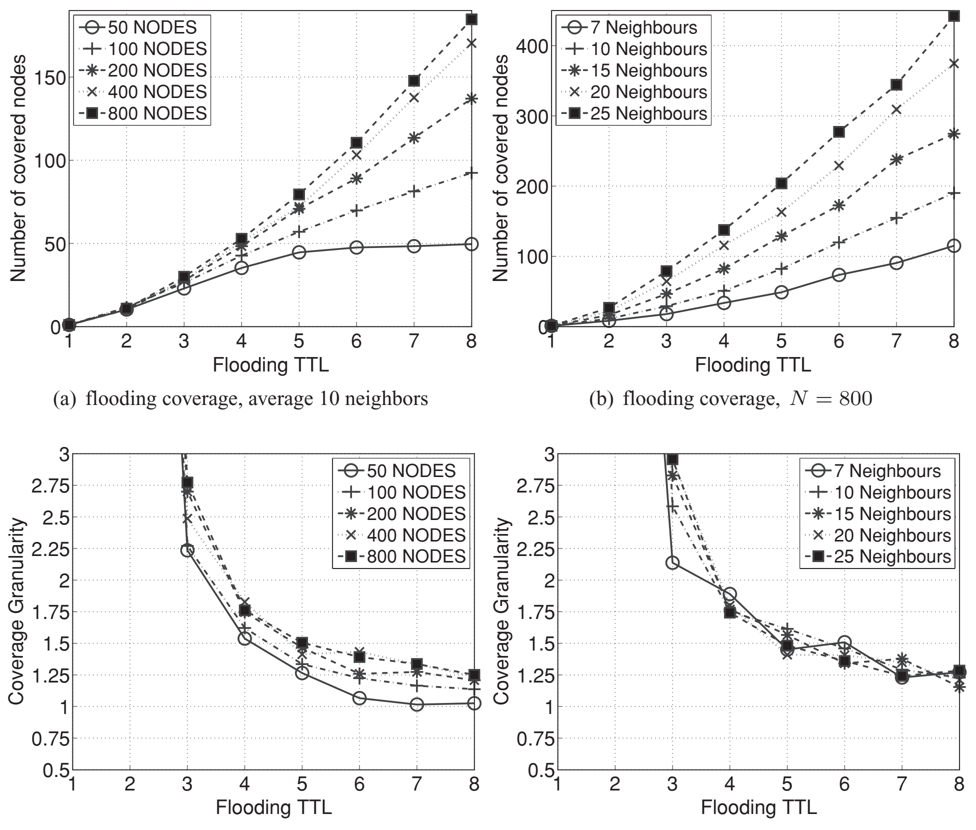
4.4 FLOODING

Another way to access a quorum is by flooding. Since in the general case flooding will reach all network nodes, which might not be necessary, a limited-scope flooding can be used instead. That is, the request is broadcasted from a given node to all its neighbors with a given Time-To-Live (TTL). Each neighbor that receives the request for the first time decreases the TTL by one, and if the result is larger than 0, rebroadcasts the message to its neighbors, etc. All nodes that receive the message are members of the quorum. FLOODING can also be used to implement `advertise` quorums, by flooding the whole network and every node deciding to take part in the `advertise` quorum with probability $|Q|/n$. To prevent multiple simultaneous broadcast transmissions from colliding, we use a random jitter [IETF Mobile Ad-hoc Networks Working Group 2008] of 10 millisecond, as suggested in Broch et al. [1998].

The biggest challenge of using FLOODING is how to set the TTL in order to ensure that the message is received by k nodes, for a given k. The amount of nodes covered by flooding directly depends on the network topology and network density. Here we suggest two possible implementations. The first is based on the explicit assumption that the nodes are uniformly distributed over the network and the average density is known (this holds, e.g., in mobile networks with random movement pattern). In such a case, the value of the TTL can be approximated analytically.

However, if the topology is different or the density is unknown or may change unpredictably, a different implementation strategy, termed *expanding ring*, can be used (widely used in reactive routing protocols in MANET [Broch et al. 1998]). According to this strategy, the originating node starts a series of flooding requests with increased TTLs. In every round, the originating node should estimate the amount of accessed nodes and continue until roughly $|Q|$ different nodes have been accessed. The counting can be implemented by nodes sending back acknowledgments. In order to reduce the excessive number of acks a number of techniques can be used. One such technique combines the acks from different nodes along the ack reverse path. Another technique is for nodes to ack probabilistically and for the originating node to estimate the amount of accessed nodes based on the number of received acks and the reply probability. Expanding ring is a robust implementation that ensures that the required number of nodes are accessed on any topology. This robustness, however, comes at an increased communication cost.

The main drawback of flooding is the inability to have a fine-grain control over the exact number of nodes that receive the message (this is a shortcoming of both techniques described earlier). This is captured by the notion of *Coverage Granularity* (CG). Coverage granularity measures the difference in the flooding coverage when increasing the TTL by one. That is, $CG(i) = \frac{N_i}{N_{i-1}}$, when N_i is the expected number of nodes that are covered by flooding with TTL i [Keidar and Melamed 2006]. $N_1 = 1$ (only the originating node). A small CG allows a more adaptive flooding that adjusts to varying densities and failures, for example, by increasing the TTL by one in the expanded ring implementation. Figure 5 depicts simulation results of TTL influence on the flooding coverage. We can



(a) flooding coverage, average 10 neighbors

(b) flooding coverage, $N = 800$

(c) coverage granularity, average 10 neighbors

(d) coverage granularity, $N = 800$

Fig. 5. Flooding coverage for varying network sizes and densities. (a) and (b) number of nodes covered by flooding as a function of Time-To-Live (TTL). (c) and (d) coverage granularity: $CG(i) = \frac{N_i}{N_{i-1}}$. Flooding has high CG for small (effective) TTLs.

see the substantial growth in the number of covered nodes when increasing TTL. $CG(3)$ is always above 2, $CG(4)$ and $CG(5)$ is between 1.75 and 1.25 for different network sizes. When varying network density, CG is approximately the same for all densities and is around 1.75 for TTL 4. Such a big flooding coverage granularity impedes its efficient use.

An additional disadvantage of FLOODING is that it does not posses the *early halting* property; there is no way to stop the flooding from expanding before the TTL expires. Another drawback for location services is the numerous number of replies that will be sent back to the starting node, which also increases the communication cost. Last but not least, broadcasting in wireless networks has a number of network-level disadvantages. For example, in 802.11 WiFi networks [IEEE Computer Society 2007] broadcasting is less reliable since nodes do not acknowledge broadcast messages; broadcasts are usually sent at the lowest possible rate of 1 or 2 Mbps since low rates enable a better signal

Advertise	RANDOM				FLOODING		PATH
Lookup	RANDOM	RANDOM-OPT	PATH	FLOODING	PATH	FLOODING	PATH
Advertise Cost	$\frac{n}{\sqrt{\ln(n)}}$				Combined Cost	Combined Cost	Combined Cost Lower bound*: $\frac{n}{\ln(n)}$
Lookup Cost	$\frac{n}{\sqrt{\ln(n)}}$	$\sqrt{n\ln(n)}$	$\sqrt{n}$	$\sqrt{n}$	n	n	Simulations: n

* lower bound is based on the crossing time

Fig. 6. Asymptotic comparison of combinations of different access strategies for a target quorums size $|Q| = \Theta(\sqrt{n})$, for random geometric graphs.

capture, which is especially important due to the lack of acks; broadcast is less energy efficient than sending point-to-point messages. The 802.11 MAC PSM protocol can put nodes to sleep, which can significantly save energy. However, PSM is disabled when nodes communicate by broadcasts.

4.5 RANDOM-OPT Access Strategy

Yet another way to access a quorum is by optimizing the RANDOM strategy. As discussed before, RANDOM either utilizes routing or sampling. In both cases, RANDOM sends messages that pass through intermediate nodes, without making an efficient use of these nodes. On the other hand, the RANDOMO-OPT strategy adds these intermediate nodes to the quorum.

Whenever a quorum access message passes through an intermediate node p, the networking layer of this node can pass this message to the location service. The location service will either perform a local lookup in case it is a `lookup` access or store the advertisement in case of an `advertise` access. In the former case, if the data is found at node p, p can respond immediately to the originator and instruct its own networking layer not to forward the lookup request any further.

The benefit of this approach is that on average, the same quorum size can be accessed by explicitly contacting many fewer nodes. Since the average length of a random route in the network is $\sqrt{\frac{n}{\ln n}}$, we can issue fewer routing requests. However, these routes may not necessarily pass in different nodes. As we show by simulation in Section 8, when using the RANDOM-OPT access strategy for `lookup`, we only need to invoke the lookup request to $O(\ln n)$ random nodes instead of $O(\sqrt{n})$ nodes.

Note however, that in contrast with the RANDOM strategy, RANDOM-OPT is not guaranteed to access randomly chosen nodes and thus it cannot simply substitute RANDOM.

5. IMPLEMENTING PROBABILISTIC QUORUM SYSTEMS

As mentioned before, we can use any of the access strategies described in Section 4 to implement any of the `advertise` and `lookup` quorums and we can mix and match them.

In what follows, we present a formal evaluation of quorum systems obtained by several of these combinations. Figure 6 summarizes the various combinations of different access strategies for $|Q| = \Theta(\sqrt{n})$.

5.1 Symmetric Combination with Two RANDOM Quorums

`advertise` RANDOM, `lookup` RANDOM. This is the method of Malkhi et al. [2001]. Lemma 3.4 from Malkhi et al. [2001] states the following.

LEMMA 5.1. *Let Q_a and Q_ℓ be quorums of size $k\sqrt{n}$ each chosen uniformly at random. Then the nonintersection probability is $Pr(Q_a \cap Q_\ell = \emptyset) < e^{-k^2}$.*

Loosely speaking, quorums of size $\Theta(\sqrt{n})$ ensure intersection with good probability. It can also be easily shown (for example, by using the same argument as we do later in Lemma 5.2) that if Q_a and Q_ℓ are quorums of sizes $|Q_a|$ and $|Q_\ell|$ accordingly, each chosen uniformly at random, then $Pr(Q_a \cap Q_\ell = \emptyset) < e^{-\frac{|Q_a||Q_\ell|}{n}}$.

As we have shown previously in Section 4, for $|Q| = \Theta(\sqrt{n})$ the cost of invoking RANDOM in an ad hoc network is $\frac{n}{\sqrt{\ln n}}$ plus the cost of establishing multihop routes in case a membership service is used or $\Theta(n\sqrt{n})$ when sampling directly with random walks.

5.2 Asymmetric Combinations with One RANDOM Quorum

In what follows we provide a proof of the main result of our article: to build probabilistic biquorums it is sufficient that only one of the quorums is RANDOM, while the other quorum can be picked in an arbitrary (nonadversarial) way.

LEMMA 5.2 (MIX AND MATCH LEMMA). *Let Q_a be an* `advertise` *quorum and Q_ℓ be a* `lookup` *quorum, and let at least one of these quorums be a RANDOM quorum. Then the probability that an intersection $Q_a \cap Q_\ell$ is empty is* $\Pr(Q_a \cap Q_\ell = \emptyset) \leq e^{-\frac{|Q_a||Q_\ell|}{n}}$.

PROOF. Since the selection of a lookup quorum is performed independently of the selection of an advertise quorum, we claim that the order of the selection does not influence the intersection probability between the two quorums. We can therefore look at the quorum selection process as if the lookup quorum Q_ℓ was selected first.

Thus, suppose that a subgroup Q_ℓ of different nodes is picked out of a network of n nodes (note that the selection may be arbitrary, not necessarily uniform). Next, another subgroup Q_a of nodes is picked, while each node in Q_a is picked uniformly at random out of the network, but without repetitions. For the intersection of Q_ℓ and Q_a to be empty, we can look at this process as if the first node of Q_a is picked out of $n - |Q_\ell|$ nodes uniformly at random, the next node is picked out of $n - |Q_\ell| - 1$ nodes, etc. Hence, we have

$$\Pr(Q_a \cap Q_\ell = \emptyset) = \prod_{i=0}^{|Q_a|} \frac{n - |Q_\ell| - i}{n - i} \leq \prod^{|Q_a|} \frac{n - |Q_\ell|}{n} = \left(1 - \frac{|Q_\ell|}{n}\right)^{|Q_a|} \leq e^{-\frac{|Q_a||Q_\ell|}{n}}. \quad \square$$

The preceding result can be used to set the size of the quorums Q_a and Q_ℓ to guarantee a nonempty intersection of two quorums with a given probability. This is established next.

COROLLARY 5.3. *In order for two sets, Q_a and Q_ℓ of sizes $|Q_a|$ and $|Q_\ell|$ respectively chosen in the manner described in Lemma 5.2 to have a nonempty intersection with probability at least $1-\varepsilon$, the following must hold: $|Q_a| \cdot |Q_\ell| \geq n \ln(1/\varepsilon)$.*

PROOF.

$$\Pr(Q_a \cap Q_\ell \neq \emptyset) \geq 1 - e^{-\frac{|Q_a||Q_\ell|}{n}} \geq 1-\varepsilon \Rightarrow -\frac{|Q_a||Q_\ell|}{n} \leq \ln(\varepsilon) \Rightarrow \frac{|Q_a||Q_\ell|}{n} \geq \ln(1/\varepsilon).$$

□

To get a feel for the result, consider the example in which $1-\varepsilon = 0.9$. In this case, $|Q_a| \cdot |Q_\ell|$ must be at least $2.3n$. Thus, we can pick both $|Q_a|$ and $|Q_\ell|$ to be $\Theta(\sqrt{n})$.

Summary. An asymmetric quorum system, in which only one of the quorums has to be picked randomly while the other one can be picked in an arbitrary (nonadversarial) way, has a significant advantage over the pure RANDOM ⊗ RANDOM strategy mix. While the cost of a RANDOM access is almost linear in n (plus the cost of establishing multihop routes), the second quorum can be accessed by a considerably less expensive access strategy. In addition, a combination in which at least one quorum is RANDOM has an important practical advantage. The intersection probability does not depend on the network topology or density (Lemma 5.2 is correct on any network graph). It means that combinations with RANDOM are robust and yet efficient, since the second (nonrandom) quorum can be picked in a very efficient way.

`advertise` *RANDOM,* `lookup` *PATH.* In this case, the selection of the `advertise` quorum is performed uniformly at random, while the selection of the `lookup` quorum is performed by a random walk. As we have shown in Section 4.2, the cost of the PATH access strategy is linear in the quorum size for $|Q| = \Theta(\sqrt{n})$. This can be further optimized by using UNIQUE-PATH, which does not affect the intersection probability, but can lower the communication cost. We can also switch the `advertise` and `lookup` access strategies: use RANDOM for `lookup` and PATH for `advertise`. We explore this direction in more detail in Section 5.4.

`advertise` RANDOM, `lookup` FLOODING. Instead of using the PATH strategy used for `lookup`, we can use FLOODING. The intersection probability in these cases is similar to the ones obtained with PATH *lookup*. This is because a FLOODING access that covers (at least) k nodes, visits k different nodes, just like a random walk. However, in most cases more than k nodes will actually receive the message. Hence, while asymptotically the cost will be the same as PATH, the constants for FLOODING are higher. In addition, FLOODING does not provide early halting and sends multiple replies, which also increases the communication cost and is not energy efficient.

`advertise` *RANDOM,* `lookup` *RANDOM-OPT.* We can also use the RANDOM-OPT strategy to access the `lookup` quorum. This is expected to produce the same intersection as by the RANDOM ⊗ RANDOM mix, while sending fewer messages.

5.3 Symmetric Combinations without RANDOM Quorum

We have further explored combinations that do not use the RANDOM access strategy at all.

`advertise` *PATH,* `lookup` *PATH.* Both `advertise` and `lookup` are performed using random walks. This neither requires any sampling or membership services nor routing and thus appears very appealing. However, a deeper look reveals that this combination is less attractive than it seems, since it has high communication and memory costs. One must ensure these two random walks intersect in at least one node. For that matter we define the *crossing time* of two random walks.

Crossing time of two simple random walks in random geometric graphs. We say that two random walks cross, if there exists at least one node in which these two random walks visit during their run (not necessarily simultaneously). Therefore, crossing time captures the smallest number of steps required for two simple random walks to intersect in at least one node.

Definition 5.4. Given two random walks starting at any two nodes in the network, the *crossing time* (CRT) is the expected first time at which there is a node that was visited by the two random walks.

Formally, given two random walks X_u, Y_v, starting at u and v respectively, let $\gamma_{uv} = \min\{t : \{X_u(0), \dots, X_u(t)\} \cap \{Y_v(0), \dots, Y_v(t)\} \neq \emptyset\}$, be the first time that the two walks cross. The crossing time of a graph is defined as $\max_{uv} E[\gamma_{uv}]$. We prove the following.

THEOREM 5.5. *The crossing time of two random walks in $\mathcal{G}^2(n, r)$ is $\Omega(r^{-2})$.*

PROOF. Divide the unit square of $\mathcal{G}^2(n, r)$ into $\lfloor \frac{1}{r} \rfloor^2$ bins of size $r \times r$ each. Each bin is indexed by a column i and a row j, where $0 \leq i, j \leq \lfloor \frac{1}{r} \rfloor$. Now look at the projection of the walk on the columns of the bins, when the walk is at column i it can only move to a bin at columns $i-1, i+1$ or to stay at column i. This resembles a simple random walk on a line with some additional probability to stay at the same node (i.e, self-loop).

Take two nodes, u at column 0 and v at column $\lfloor \frac{1}{r} \rfloor$. For the two walks to cross, at least one walk has to reach the column in the middle. This will amount to the expected time it takes to a simple random walk on the line to move from 0 to $\lfloor \frac{1}{2r} \rfloor$, which is known to be $\Omega(r^{-2})$ [Lovász 1993]. □

Note that when $r = \Theta(\sqrt{\frac{\log n}{n}})$ (the minimal radius that ensures connectivity), the crossing time of $\mathcal{G}^2(n, r)$ is at least $\Omega(\frac{n}{\log n})$. Thus, if both `advertise` and `lookup` quorums are accessed by the PATH strategy, at least one of them will incur a communication cost that is almost linear in n. Recall that this is a lower bound, and, as a matter of fact, our simulation results in Section 8 indicate that *both* `advertise` and `lookup` need to be of that order, even if we use a unique random walk. In addition, there is an increased storage cost. The `advertise` random walk has to publish the data in every node it visits and if the length of this random walk is $\Omega(n/\log n)$, accessing the `advertise` quorum will incur almost linear storage.

Another shortcoming of the PATH $\otimes$ PATH combination is that the exact value of the crossing time depends on the network density and is thus hard to set without overestimating it or running in danger of not ensuring the desired intersection probability.

`advertise` *UNIQUE-PATH,* `lookup` *UNIQUE-PATH.* Instead of using a simple RW we can use the unique RW. As before, the sizes of 2 quorums should be set to guarantee that the two quorums intersect. For example, setting $|Q_a| + |Q_\ell > n$ provides such a guarantee. In such a case our result about partial cover time from Section 4.2 can be used as an estimate of the message complexity of this combination.

`advertise` *FLOODING,* `lookup` *FLOODING.* Here, in the same manner as with random walks, one must ensure that the two sets of nodes accessed by two flooding requests intersect. It is thus clear that the combined cost of both flooding requests is linear with n.

5.4 Deriving the Optimal Cost for Asymmetric Combinations

An ability to build quorum systems while using different access strategies and different quorum sizes raises the following interesting question. What is the best strategy mix and what is the best relative size of the `lookup` and `advertise` quorums that guarantee a given intersection probability with an optimal minimal cost? The answer to this question depends on the usage pattern of `lookup` versus. `advertise`. Intuitively, if `lookup` is used more frequently than `advertise`, one would suggest to optimize the `lookup` access strategy, by using a smaller `lookup` quorum size. The question we raise is how much exactly to optimize?

To this end, we consider the total cost of accessing all quorums. This total cost depends on the size of each quorum, the cost of accessing a single node in this quorum, and the relative ratio of requests to access `lookup` versus `advertise`. More formally, we define the function *Cost* to be the cost of accessing a node or a set of nodes. The cost function could reflect the number of messages or any other measure. $Cost(Q_a)$ is the cost of accessing an `advertise` quorum and $Cost_a = \frac{Cost(Q_a)}{|Q_a|}$ is the average cost of accessing a single node in the `advertise` quorum. $Cost(Q_\ell)$ and $Cost_\ell$ are defined similarly for the `lookup` quorum.

We assume a parameter τ, which is the network-wide ratio between the number of times `lookup` is being accessed versus `advertise`.

LEMMA 5.6. *The optimal ratio between the sizes of* `lookup` *and* `advertise` *quorums, which minimizes the total cost of accessing all* `lookup` *and* `advertise` *quorums, is*

$$\frac{|Q_\ell|}{|Q_a|} = \frac{1}{\tau}\frac{Cost_a}{Cost_\ell}.$$

PROOF. Define `#advertise` to be the total number of advertisements issued by all nodes during the considered period of time and by `#lookup` the total number of lookups. The total cost is

$$TotalCost = \texttt{\#advertise} \cdot |Q_a| \cdot Cost_a + \texttt{\#lookup} \cdot |Q_\ell| \cdot Cost_\ell.$$

We are subject to the constraint that $|Q_a| \cdot |Q_\ell| = n \ln(1/\varepsilon)$ and are given $\tau = \frac{\texttt{\#lookup}}{\texttt{\#advertise}}$. Therefore,

$$TotalCost = \frac{\texttt{\#lookup}}{\tau} \cdot \frac{n \ln(1/\varepsilon)}{|Q_\ell|} \cdot Cost_a + \texttt{\#lookup} \cdot |Q_\ell| \cdot Cost_\ell.$$

The minimal *TotalCost* is achieved when $|Q_\ell| = \sqrt{\frac{n \ln(1/\varepsilon) Cost_a}{\tau Cost_\ell}}$ (obtained by derivation of *TotalCost*). Thus, $\frac{|Q_\ell|}{|Q_a|} = \frac{|Q_\ell|^2}{n \ln(1/\varepsilon)} = \frac{1}{\tau} \frac{Cost_a}{Cost_\ell}$. $\square$

Consider an example in which $\tau = 10$ (there are 10 times more lookups than advertisements). Let `advertise` quorum be accessed by a RANDOM strategy and `lookup` by a UNIQUE-PATH strategy. $Cost_a = D$ (the diameter of the network) and $Cost_\ell \approx 1$ (as we have shown in Section 4.2). For a network with $D = 5$, we must pick $\frac{|Q_\ell|}{|Q_a|} = \frac{5}{10} = 1/2$. Thus, to yield the minimal total access cost, the size of the `advertise` quorum should be twice the size of the `lookup` quorum.

The parameter τ is a global network parameter. It might be known a priori to all nodes or configured based on common usage patterns (such as the distribution of advertisements versus lookups in file sharing P2P applications). In case τ is not known and cannot be assumed, it can be dynamically estimated based on the usage statistics.

Notice that even if τ is estimated wrongly or changes over time, the correctness/safety of the quorum system (intersection probability) will not change. A wrong τ estimation will result in picking the strategy mix that is not optimal with respect to the message overhead. One way to deal with τ changes is to adapt and refresh the quorum system, similar to what we do to handle churn in Section 6.1.

6. MAINTENANCE: HANDLING FAILURES, DYNAMISM, AND MOBILITY

Probabilistic constructions are inherently very suitable to handle dynamic environments, such as networks with frequent node joins and failures or mobility. This is because they do not rely on strict determinist sets of nodes which are costly to update and reconfigure. In this section we describe some of the formal properties of probabilistic quorums with respect to this dynamism, as well as provide additional implementation details to improve dynamism handling even further.

6.1 Handling Churn

Churn is caused by frequent joins and leaves/failures of nodes. As mentioned in Section 3, the resilience of the quorum system to failures is measured by fault tolerance and failure probability. The fault tolerance of a probabilistic quorum system with quorum sizes of $\sqrt{n}$ is $\Omega(n)$. Similarly, the failure probability of a probabilistic quorum system of size $k\sqrt{n}$ is $e^{-\Omega(n)}$ for all $p \leq 1 - \frac{k}{\sqrt{n}}$. However, in ad hoc networks, we must also require that the network remains connected. We discuss the necessary condition for connectivity next.

Connectivity in face of failures. The connectivity of $\mathcal{G}^2(n, r)$ was extensively studied in the context of the minimal transmission power necessary to ensure that, with high probability, a given ad hoc network graph is still connected as the number of nodes in the network grows to infinity. Gupta and Kumar [1998] have shown that if n nodes are placed on a unit disk and each node transmits at a power level that covers an area of $\pi r^2 = \frac{\log n + c(n)}{n}$, then the resulting network is asymptotically connected with probability one, if and only if $c(n) \to \infty$ as $n \to \infty$. In Panchapakesan and Manjunath [2001], the authors obtain a similar result when nodes are distributed in the unit square $[0, 1]^2$. The previous result implies that if the transmission range r is set such that $r = \sqrt{\frac{C \ln n}{\pi n}}$ for $C > 1$, the network is connected with high probability. Such a value of r implies an average number of neighbors, d_{avg}, which is $d_{\text{avg}} = \pi r^2 n = C \ln n$.

With failures, one would like to know how many failures leave the network connected. We look at a network with fixed r and assume a failure model in which individual nodes crash independently with fixed probability. In such a model, after i nodes fail, the remaining network forms a random geometric graph, $\mathcal{G}^2(n-i, r)$. This network remains connected if $n-i$ satisfies the necessary connectivity condition, namely, $r \geq \sqrt{\frac{\ln(n-i)}{\pi(n-i)}}$. For example, in the network of 1000 nodes the minimal d_{avg} that guarantees connectivity is 7. Thus, if the initial density is $d_{\text{avg}} = 14$, this network can withstand a failure of up to half of the nodes in the network.

Degradation rate. Degradation rate captures the probability that two quorums accessed at different times will intersect despite the fact that between these two accesses some nodes have crashed or joined. We analyze the degradation rate as a function of the percentage of the network that changed (percentage of crashed or joined nodes). We calculate the probability of an intersection of a `lookup` quorum with a previously established `advertise` quorum. Denote by $Q_a(t)$ the live nodes of a given `advertise` quorum at time t. $Q_a(0)$ is the initial `advertise` quorum at the time it was established (before the churn started) that guarantees intersection with at least $1 - \varepsilon$ probability. $Q_\ell(t)$ is a `lookup` quorum at the moment the access is being issued (`lookup` quorum accesses only live nodes at time t, so they are not affected by failures). $n(t)$ is the network size at time t (at the time of a `lookup` access). We denote the nonintersection probability at time t by $\Pr(miss(t))$. By definition, $\Pr(miss(0)) \leq \varepsilon$.

$$\Pr(miss(t)) = \Pr(Q_a(t) \cap Q_\ell(t) = \varnothing) \leq e^{-\frac{|Q_a(t)||Q_\ell(t)|}{n(t)}}$$

We separate our discussion into cases of failures only, joins only, and both, and explore each under different churn rates. Note that when nodes fail or join, $n(t)$ changes as well. We can furthermore separate the discussion into two additional categories: whether the size of the `lookup` quorum, $|Q_\ell(t)|$, is adjusted to $n(t)$ or not. Practically, this means that the data location service that uses quorums can periodically estimate the network size $n(t)$ and adjust $|Q_\ell(t)|$ dynamically to its size (e.g., $|Q_\ell(t)| = C\sqrt{n(t)}$), or keep $|Q_\ell(t)|$ constant ($|Q_\ell(t)| = |Q_\ell(0)|$) until the next time the whole quorum system is refreshed (as discussed next).

(1) *Failures only*. We assume nodes crash independently, with some fixed probability. In such a case $n(t) = (1 - f)n(0)$ and $|Q_a(t)| = (1 - f)|Q_a(0)|$, while f is the fraction of failed nodes. Thus,

$$\Pr(miss(t)) \le e^{-\frac{|Q_a(t)||Q_\ell(t)|}{n(t)}} = e^{-\frac{|Q_a(0)|(1-f)|Q_\ell(t)|}{(1-f)n(0)}} = e^{-\frac{|Q_a(0)||Q_\ell(t)|}{n(0)}}.$$

(a) If $|Q_\ell(t)| = |Q_\ell(0)|$ ($|Q_\ell(t)|$ is kept constant and is not adjusted to $n(t)$), then $\Pr(miss(t)) = \Pr(miss(0))$ (it does not change)! This is despite the fact that a fraction f of the network, including the advertisement nodes, has failed.

(b) If $|Q_\ell(t)|$ is adjusted to $n(t)$, for example, $|Q_\ell(t)| = C\sqrt{n(t)}$ then

$$\begin{aligned}\Pr(miss(t)) &\le e^{-\frac{|Q_a(0)||Q_\ell(t)|}{n(0)}} = e^{-\frac{|Q_a(0)|C\sqrt{n(t)}}{n(0)}}\\ &= e^{-\frac{|Q_a(0)|C\sqrt{(1-f)n(0)}}{n(0)}} = e^{-\frac{|Q_a(0)|\sqrt{(1-f)}|Q_\ell(0)|}{n(0)}}\\ &\le \varepsilon^{\sqrt{1-f}}.\end{aligned}$$

(2) *Joins only*. In such a case $n(t) = (1 + f)n(0)$ and $|Q_a(t)| = |Q_a(0)|$, while f is the fraction of joined nodes.

$$\Pr(miss(t)) \le e^{-\frac{|Q_a(t)||Q_\ell(t)|}{n(t)}} = e^{-\frac{|Q_a(0)||Q_\ell(t)|}{(1+f)n(0)}}$$

(a) If $|Q_\ell(t)| = |Q_\ell(0)|$, then $\Pr(miss(t)) \le e^{-\frac{|Q_a(0)||Q_\ell(t)|}{(1+f)n(0)}} = e^{-\frac{|Q_a(0)||Q_\ell(0)|}{(1+f)n(0)}} \le \varepsilon^{\frac{1}{1+f}}$.

(b) If $|Q_\ell(t)| = C\sqrt{n(t)}$, then $\Pr(miss(t)) \le e^{-\frac{|Q_a(0)|C\sqrt{n(t)}}{(1+f)n(0)}} = e^{-\frac{|Q_a(0)|C\sqrt{(1+f)n(0)}}{(1+f)n(0)}} \le \varepsilon^{\frac{1}{\sqrt{1+f}}}$.

(3) *Both joins and failures*. If the same amount of nodes that have failed have also joined, then $n(t) = n(0)$, $|Q_a(t)| = (1 - f)|Q_a(0)|$, and $|Q_\ell(t)| = |Q_\ell(0)|$.

$$\Pr(miss(t)) \le e^{-\frac{|Q_a(t)||Q_\ell(t)|}{n(t)}} = e^{-\frac{(1-f)|Q_a(0)||Q_\ell(0)|}{n(0)}} \le \varepsilon^{1-f}$$

Figure 7 illustrates the evolution of the intersection probability as a function of the churn rate for all cases. We can see that if we keep using the same `lookup` quorum size while nodes fail, the intersection probability remains the same. This result indicates a remarkable resilience of probabilistic biquorum systems to failures. If there are no new joins, then the intersection probability stays the same despite failures! In the case of joins, a biquorum system can withstand a churn of a linear fraction of nodes without a significant intersection deterioration. For example, when starting with an initial intersection probability of 0.95, after 30% of the network changed (30% of the nodes have failed and

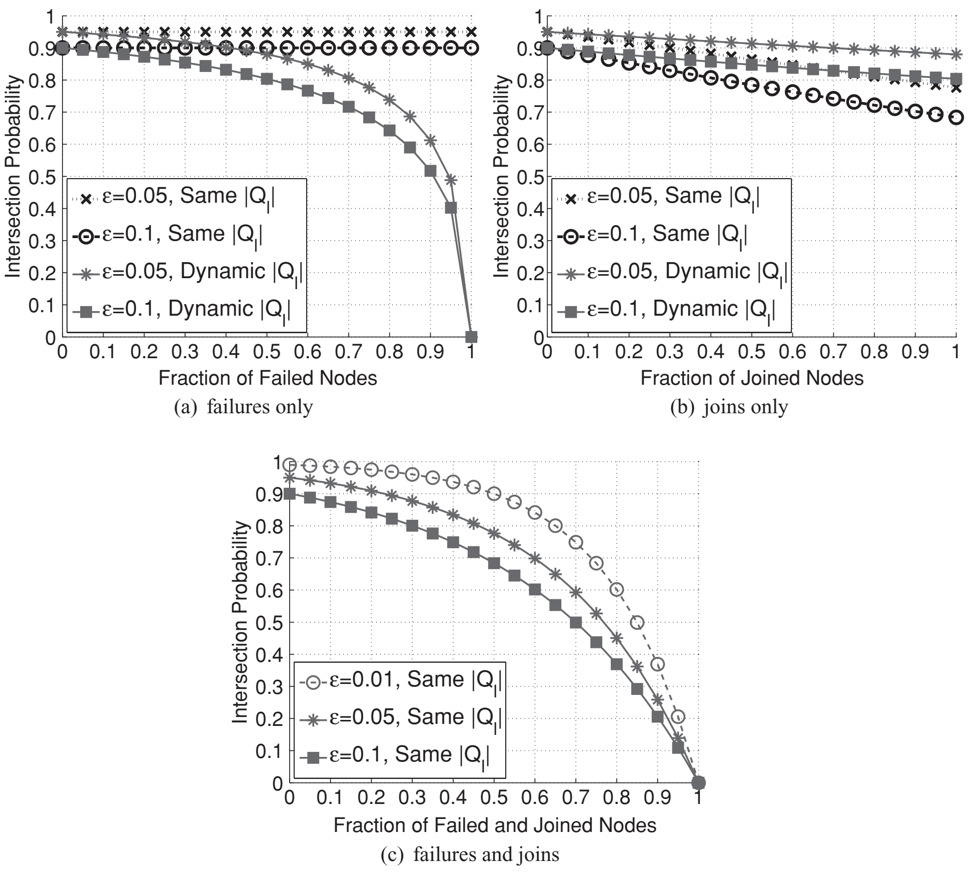


Fig. 7. The degradation of the intersection probability as a function of f: the fraction of crashed and joined nodes. ε is the initial nonintersection probability.

new ones have joined), the intersection probability deteriorates to only slightly below 0.9.

Handling quorum degradation. In the case of deterministic quorums, recovering from quorum degradation requires both reconfiguring the quorum system (finding a new quorum set that will replace the previous one) and refreshing its contents, in order to ensure data continuity, as has been done in Lynch and Shvartsman [2002] and Abraham and Malkhi [2003]. In the case of probabilistic quorums, there is no need to reconfigure the system after failures in order to ensure quorum liveness. All that is needed is to refresh the quorum system, such as by readvertising every data item to ensure data continuity. The frequency of this readvertising is determined by the degradation rate. Consider the example depicted in Figure 7(c). Suppose the minimum accepted intersection probability of a given system is 0.9, the intersection probability before the churn started was 0.95, and the time it takes 30% of the nodes to change is one day. Then in this example, every data item should be refreshed once a day.

Notice that the way the data should be refreshed depends on the intersection requirement, which is driven by the data consistency model and the service usage pattern. For the relaxed intersection requirement, described in Section 2.5, readvertising is enough. On the other hand, implementations of general data types (or services) tend to use version numbers (or timestamps) that are typically stored alongside the value of each data item, in order to ensure that a new value cannot be overwritten by an older one. In particular, for linearizable read/write objects [Herlihy and Wing 1990] (shared memory atomic registers, lock servers, etc. [Attiya and Welch 1998; Lynch 1996]), each update operation must first read the current version of the data via a `lookup` quorum in order to learn the most recent version number and only then write the new value to an `advertise` quorum [Attiya et al. 1995; Gramoli 2007]. Consequently, a refresh of the quorum system must act similarly, that is, first read the current value of the data via a `lookup` quorum and next write it back to an `advertise` quorum [Friedman et al. 2005; Gramoli 2007]. Certain optimizations can be applied, but these are often highly application and consistency model specific (and hence outside the scope of the article).

6.2 Handling Mobility: Network and Application Adaptation

In ad hoc networks, our construction should also handle mobility. If mobility maintains the uniform distribution of nodes, then it does not impact the intersection probability. Thus, in these cases, nothing need be done. However, if mobility substantially skews the structure of the network, then refreshing by readvertising is required. The rate of refreshing in such cases depends on the exact mobility model.

Another important aspect of mobility is network-level adaptation. Consider, for example, a random walk implementation. The next hop along the random walk is picked randomly out of each node's direct neighbors, while the neighbors list is constructed by a heartbeat like mechanism [hoc Networks Working Group 2007]. However, due to mobility, the neighbors set can change rapidly, which might result in attempting to forward the random walk to a nonneighbor or even a failed node. Another example is RANDOM quorum with membership implementation. The accessed node can leave the network or simply fail. However, this might not be immediately indicated by the membership service. These examples demonstrate a need for: (1) reliable indication of failures to access nodes; (2) an ability to dynamically recover from such failures.

Network-level notifications. Indicating failures in accessing quorums can be achieved, for example, by applying end-to-end reliability [Saltzer et al. 1984] design: the accessed node will send an ack to the originator. This, however, has a significant cost of doubling the traffic and may also introduce some unnecessary latency. Instead, we suggest to use low-level network notifications, in a cross-layer design. For example, if the MAC protocol fails to receive an ack for a unicast transmission (after multiple attempts, default 7 in 802.11 MAC [IEEE Computer Society 2007]), instead of simply dropping the packet, it can signal a higher layer about this failure. This notification should be propagated in the networking stack all the way to the application, allowing it to act accordingly.

Another example is a failure of the routing protocol to establish a routing path. This will happen if the routed-to node has left the network. We have also observed such behaviors with very fast mobility, so this problem is real. In such a case, instead of silently dropping the message, routing will notify a higher layer which will propagate this notification to the application.

Application adaptation to network failures. When the application is notified about a failure of a specific message, it should employ some adaptation mechanism. Simply resending the same message again to the same destination is a bad choice, since it will fail to arrive again with high probability, consuming valuable network resources. Instead, in the case of RANDOM quorum we could pick another random node to access. In the case of random walks, we can use the *RW salvation* technique of Bar-Yossef et al. [2008] to prevent dropping of random walk messages. If node v does not succeed to forward a random walk message to the neighbor chosen in a given step (did not receive a MAC-level ack), v makes a new attempt to send this message to another random neighbor within the same step. This is especially useful in mobile networks which experience frequent breakages of neighborhood connections. We demonstrate the usefulness of this technique in Section 8.

Another example is the reply message of the random walk; according to the PATH strategy, the random walk stops at the first intersecting node and then sends the reply back to the looking node, following the reverse random walk path recorded in the random walk message. When following the reverse path in a mobile network, there is a nonnegligible chance that the reply messages will be dropped by at least one hop. Such dropping becomes more intense with growing random walk length and with increased mobility. We thus suggest to use the following *reply-path local repair* technique to combat fast mobility. When node v wishes to pass the reply message to node u, which is the next hop along the reverse path, it first sends a direct unicast MAC message to u without relying on routing. If v's MAC indicates a failure to send this message (v did not receive a MAC-level ack from u, probably since u moved out of v's range), v gives up on sending the message to u and tries to pass it to the next node along the path (node w). Since w is not necessarily v's neighbor, v sends this message with the help of routing. However, we do not want to use routing too aggressively. A naïve use of routing might cause a network-wide flooding, which will happen if the routing algorithm will search the path to w and w has also moved far from v. To prevent this costly effect, we limit the routing scope with TTL 3. Thus, routing will not search the path more than 3 hops away from v. The value of 3 was chosen as a good trade-off between the amount of exceeding traffic generated by routing search packets and the probability of receiving the reply. In the case routing fails to find the path to w (which is indicated at v by a routing-level notification), v attempts the next hop along the reverse path in the same manner, also using TTL 3. If w is the last hop along the path, and routing with TTL 3 failed to find it, then v has no choice but to invoke routing to w with a large TTL. Another option in this case is for v to drop this reply message. As we explore in Section 8, in relatively low mobility scenarios, the local repairs are sufficient to fix temporal breakages in the reverse path.

However, in a very fast mobility scenario of 20m/s (a VANET scenario), local repairs alone are not enough and occasional global routing is inevitable.

6.3 Network Size Estimation

In order to calculate the quorum size in all our access strategies, the number of nodes in the network n must be known. There are several methods for obtaining a loose upper bound on the network size, such as Feige [1996]. Notice that overestimating the network size will not hurt the intersection probability and will only incur additional communication cost. Once we have such a loose upper bound, a technique in which the nodes count the number of collisions between random walks and estimate n in the manner similar to the birthday paradox principle can be applied. This technique was previously suggested in Massoulie et al. [2007] and Bar-Yossef et al. [2008].

7. ADDITIONAL OPTIMIZATIONS

7.1 Location Service-Specific Optimizations

Typically in data location services the mapping of an object to its home node remains valid for a long time. In fact, in some situations, once the object is mapped, its location never changes. For such services two additional optimizations are possible.

Early halting. For implementing the relaxed intersection property of Section 2.5, whenever the searched data is found, the lookup quorum access can terminate early and return this result without accessing the entire quorum. We call the ability of an access strategy to terminate early without incurring the overhead of a full quorum access an *early halting*. The PATH access strategy indeed has this property: the moment the searched data is found a reply can be returned immediately and the random walk be stopped. As for the efficiency of this optimization, for data that has been published, early halting usually halves the length of the random walk (the random walk is expected to visit only half of the $|Q|$ nodes before terminating). We show this result in Section 8.

Caching. Caching of advertisement requests or lookup replies that pass through nodes can significantly reduce the lookup overhead. Here we distinguish between the node who is part of the `advertise` quorum, which we call an *owner* of the mapping, and the other nodes that happen to cache it, which we call *bystanders*. Once a node runs low on memory, it can forget all entries for which it is a bystander, but it is supposed to maintain the entries for which it serves as an owner.

With the preceding two optimizations, and especially when using the PATH strategy, lookup requests for popular data items can terminate much faster.

7.2 Random Walk Optimizations

We propose two additional optimizations for random walk-based techniques. The first one is called *reply-path reduction*. It is applicable for PATH or UNIQUE-PATH `lookup` quorums and is used for reply messages. Whenever a lookup random walk hits an `advertise` quorum, a reply is sent over the reverse path of the random walk. Whenever a reply message arrives at some node v and its next hop in the reverse path is u, v checks if any of its neighbors w appear on the reverse path further after u. In the affirmative, v sends the reply message directly to w skipping u. This optimization reduces the reply length, as demonstrated by simulations.

The second optimization utilizes the broadcasting nature of ad hoc networks. Nodes can overhear messages, for example, by switching their MAC to a promiscuous mode. If a node u that hears a random walk `lookup` request passing through one of its neighbors v is part of the `advertise` quorum for this data item (u stores this item), u can send a reply immediately to v, which will stop the random walk and send the reply back to the `lookup` originator. Thus, the number of nodes covered by a random walk is significantly increased. Exploring the benefits of this technique is left for future work.

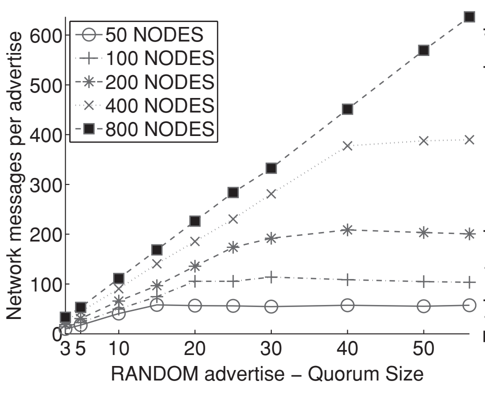
8. SIMULATIONS

We have compared our different implementation strategies by simulations under a wide range of varying conditions. In particular, we have considered the impact of the number of nodes and nodes density, mobility, churn, quorum strategy mixes, and sizes. Simulation setup is described in Section 2.4.

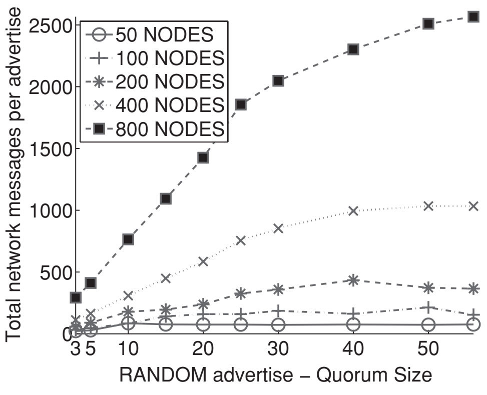
Simulation scenarios. Each simulation is comprised of two parts. In the first part, a total of 100 advertisements were performed by random nodes, each by RANDOM access to a quorum of size $2\sqrt{n}$ (except for UNIQUE-PATH `advertise` in Section 8.5). RANDOM access was based on a membership service. In the second part 1000 lookups were performed (by 25 random nodes, each making 40 lookups). `lookup` quorum was accessed by 4 different methods: RANDOM, RANDOM-OPT, UNIQUE-PATH and FLOODING. On a hit, a node sends a reply to the node that originated the lookup request. In case of RANDOM and RANDOM-OPT the reply was sent using routing, while in UNIQUE-PATH and FLOODING it was sent over the reverse path of the lookup message, thus no routing was used at all. In all simulations, the number of messages denotes network-layer messages (e.g., one application message sent to a random node that traverses a route of 4 hops is counted as 4 network-layer messages). Additional routing overhead means routing layer-specific messages, including path establishment and maintenance messages (RREQ, RREP and RERR in AODV). Hit ratio corresponds to the number of successful `lookup` quorum accesses that intersected with the corresponding `advertise` quorum. Thus, hit ratio corresponds to the intersection probability.

Each simulation lasted for 1,000 seconds (of simulation time) and each data point was generated as an average of 10 runs. Simulations started after a 200 second initialization period, which was enough to construct the membership information (in case of RANDOM quorums). Every node maintained a membership list of random, uniformly chosen, $2\sqrt{n}$ nodes.

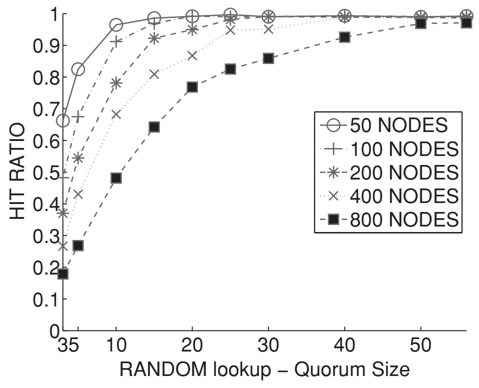
Organization. The rest of this section is organized as follows: First, we study the costs of RANDOM `advertise`. We proceed by exploring the costs of



(a) number of messages per adv



(b) num. msgs per adv + routing



(c) RANDOM lookup, Hit ratio.

Fig. 8. (a,b) Cost of RANDOM `advertise`; (c) RANDOM `lookup`. Static networks, $d_{\text{avg}} = 10$.

various combinations of `advertise` and `lookup` access strategies, each combination separately. Finally, in Section 8.8, we summarize the comparison between the various combinations based on the reported individual findings (Figures 15 and 16).

8.1 Cost of RANDOM `advertise`

Figure 8 depicts the cost (number of messages) of `advertise` by the RANDOM access strategy. This cost does not include the cost of the membership service: we assume this cost is amortized over all `advertise` accesses and is also potentially shared by other applications using the membership service (Bar-Yossef et al. [2008] includes a detailed study of random membership costs). The number of messages per `advertise` request behaves as $\frac{|Q_\ell|\sqrt{n}}{\ln n}$. The number of messages stays constant for $|Q_\ell| \geq 2\sqrt{n}$ for all networks since we use random membership of size $2\sqrt{n}$ and only those nodes were accessed for `advertise`. One can see the dramatic communication overhead increase due to routing. This is primarily due to new routes establishment and route maintenance of the AODV protocol. One has to note that the price of establishing the routes is amortized over different quorum accesses due to routes reuse and its relative part will drop in a longer run. However, in moving networks in which routes break and need to be reestablished, the price of routing remains a dominant performance factor.

8.2 RANDOM `advertise` with RANDOM `lookup` and RANDOM-OPT `lookup`

Figure 8(c) depicts the hit ratio of the RANDOM `lookup` access strategy. A hit ratio of 0.9 is achieved when the quorum size is approximately $1.15\sqrt{n}$, just as predicted by the formal analysis in Lemma 5.1. For example, for a network of 800 nodes, a quorum size of 33 achieves 0.9 hit ratio. The cost to access 33 random nodes for `lookup` is the same as the cost to access them for `advertise`, and thus can be deduced from Figure 8(a). The `lookup` quorum was accessed in parallel, thus we do not see the potential advantage of early halting. If we were to access it serially, we would see a two times reduction in the number of accessed lookup nodes, at the cost of increased latency.

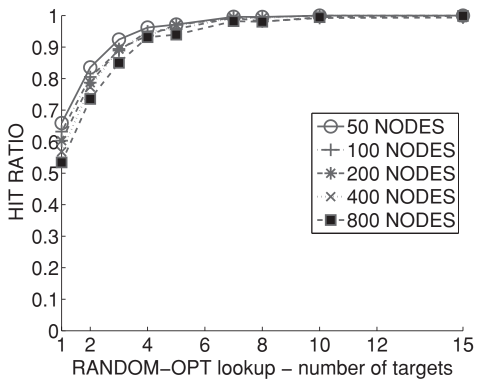
As for the RANDOM-OPT strategy (Figure 9), the hit ratio of 0.9 is achieved when starting somewhat between $X = \ln(n)$ and $X = \sqrt{\ln(n)}$ messages to random targets. Due to the cross-layer optimization of RANDOM-OPT, in which a local lookup is performed in every node through which a message passes, the actual accessed quorum size is $X\sqrt{\frac{n}{\ln n}} \approx \sqrt{n \ln n}$. Thus, this optimization reduces the communication cost significantly compared to RANDOM. For example, in a static network of 800 nodes sending 4 `lookup` requests to random nodes achieves a hit ratio of above 0.9 at the cost of less than 40 network messages, which is around $1.4\sqrt{n}$. The routing price of RANDOM-OPT is also much less than with RANDOM, since it uses fewer multihop routes. Still, the additional cost of routing is high, which makes this method inefficient compared to the UNIQUE-PATH and FLOODING strategies that are explored next.

In mobile networks, the hit ratio of RANDOM-OPT is only slightly smaller than the hit ratio achieved in static networks for the same quorum size (see Figure 9(b) versus Figure 9(a)).[5] This is because about 10% of the messages are lost due to mobility, mainly influencing the replies. The number of messages also increases (see Figure 9(e) versus Figure 9(c)). Generally speaking, the average path length in mobile networks tends to be longer than in static networks, mainly due to stale neighborhood information used by routing to find routes. When a message follows a wrong path, it takes it longer to finally get to its destination. The routing price in mobile networks also dramatically increases (see Figure 9(f) versus Figure 9(d)).

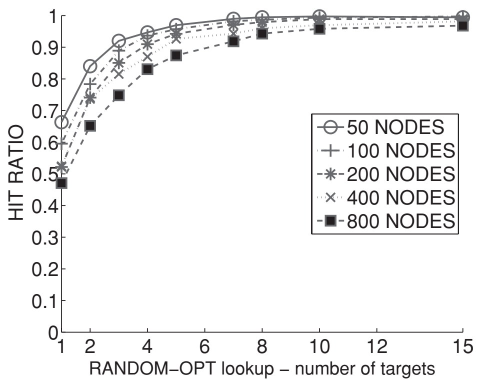
8.3 RANDOM `advertise` with UNIQUE-PATH `lookup`

Figure 10 depicts the performance of the UNIQUE-PATH strategy in mobile networks with speeds ranging between 0.5m/s and 2 m/s, which corresponds to walking speed. It performed identically in mobile and static networks and

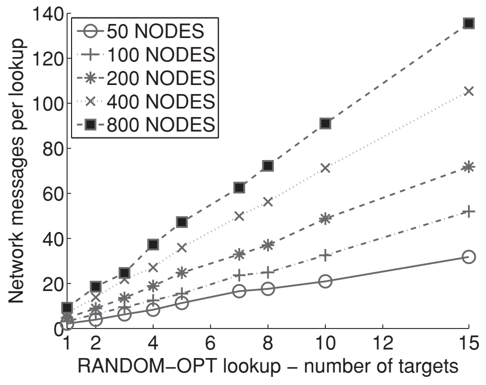
[5]The results for RANDOM `lookup` in mobile networks had the same tendency and are thus not depicted for brevity.



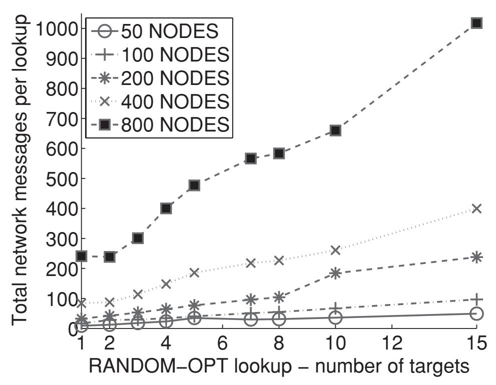
(a) hit ratio. static



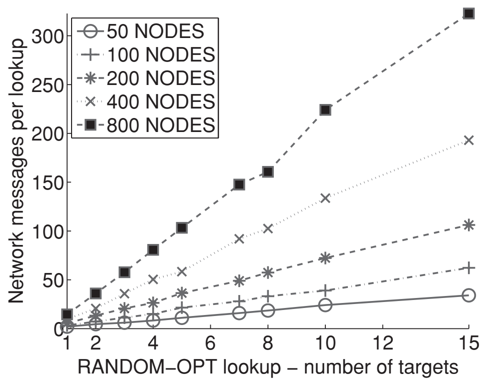
(b) hit ratio. mobile



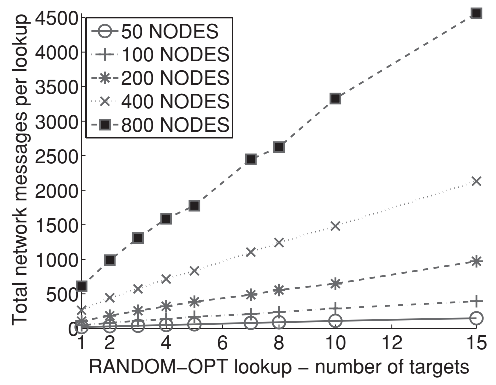
(c) num. msgs per lookup. static



(d) num. msgs per lookup + routing. static

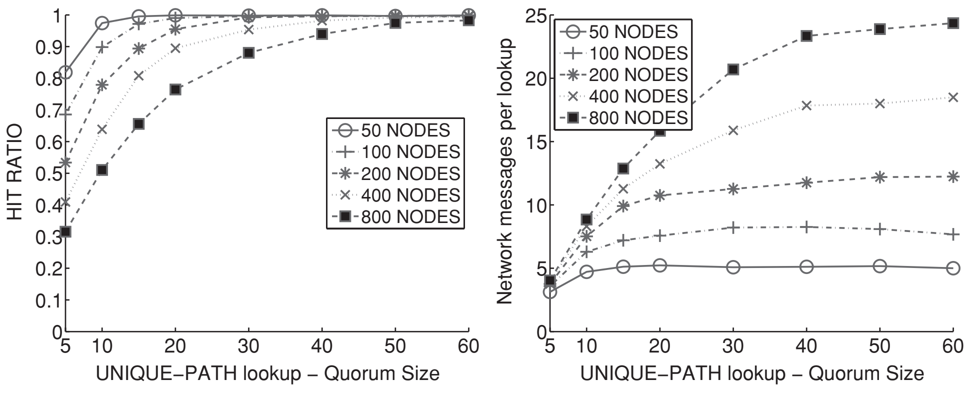


(e) num. msgs per lookup. mobile



(f) num. msgs per lookup + routing. mobile

Fig. 9. RANDOM `advertise`, RANDOM-OPT `lookup`, static and mobile networks, $d_{\text{avg}} = 10$.



(a) hit ratio. mobile (b) number of messages per lookup. mobile

Fig. 10. RANDOM `advertise`, UNIQUE-PATH `lookup`. Mobile networks, $d_{\text{avg}} = 10$.

thus we depict only the mobile case here (further mobility impact is discussed in Section 8.6). A hit ratio of 0.9 is achieved when the target quorum size (the number of nodes that need to be covered by random walk) is $\sim 1.15\sqrt{n}$, thus validating our analysis in Lemma 5.2 and testifying that a nonrandom choice of the `lookup` quorum results in the same intersection probability as a random one. The most interesting fact about the UNIQUE-PATH strategy is the extremely small number of messages it requires. One would expect that, due to the reply message, the total number of messages to access a quorum of size $|Q_\ell|$ will be $2|Q_\ell|$. Surprisingly, accessing a target quorum of size $|Q_\ell|$ requires fewer than $|Q_\ell|$ messages, including the reply message!

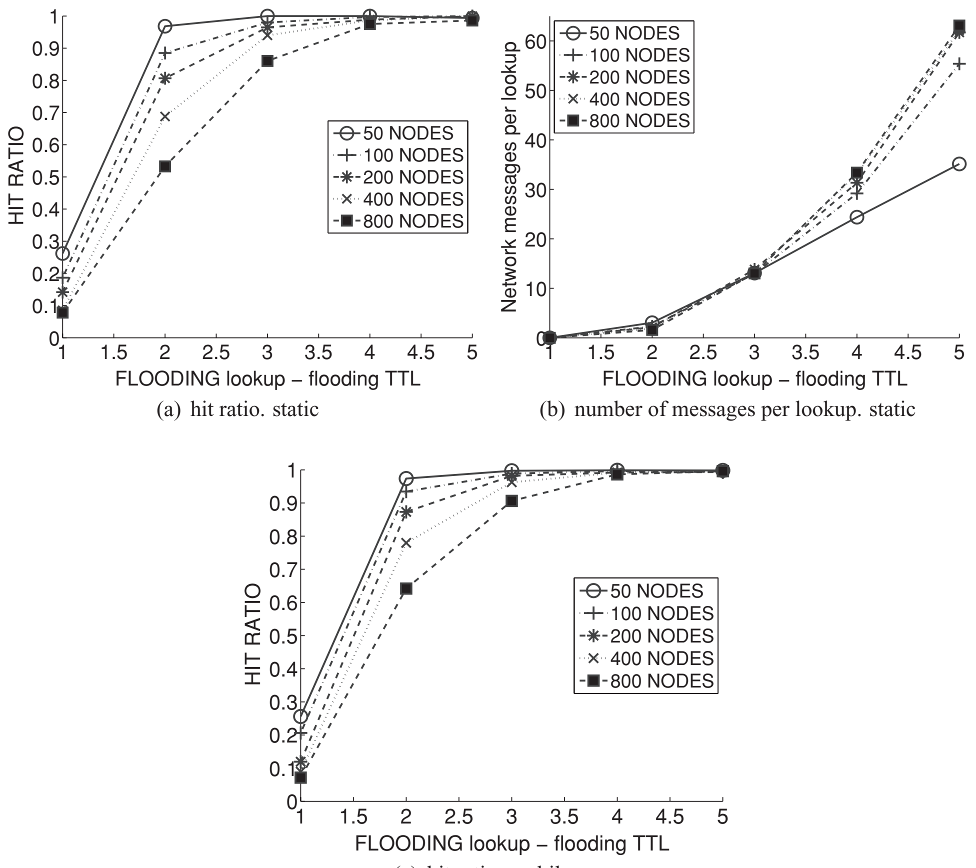
This happens due to the early halting. When a quorum of target size $|Q_\ell|$ is accessed, the first hit occurs at approximately half of the way. Thus, an average of $|Q_\ell|/2$ messages are sent until this hit. The reply follows the reverse path, but due to the *path reduction* optimization described in Section 7, the reply-path length is usually shorter. In addition, the originator of the lookup includes itself in the `lookup` quorum, which further reduces the number of messages by one.

A big advantage of the random-walk-based strategies is that they do not require multihop routing. Hence, their performance does not deteriorate by mobility, making them well suited for dynamic mobile networks. More details about the mobility impact and the usage of the random walk salvation and reply-path local repair techniques are discussed in Section 8.6.

Perhaps the biggest advantage of the RANDOM x UNIQUE-PATH mix is that the same intersection is guaranteed on any network topology and density. All that is needed is to access $|Q_\ell|$ different nodes by the random walk and this number does not depend on network characteristics.

8.4 RANDOM `advertise` with FLOODING `lookup`

Figure 11 depicts the performance of the FLOODING access strategy. The hit ratio grows super linearly with TTL. For example, for 800 nodes, a hit ratio of

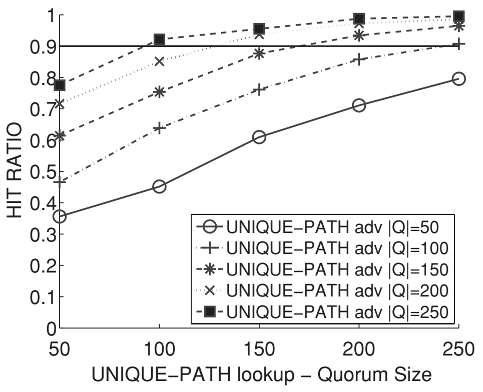


(a) hit ratio. static
(b) number of messages per lookup. static
(c) hit ratio. mobile

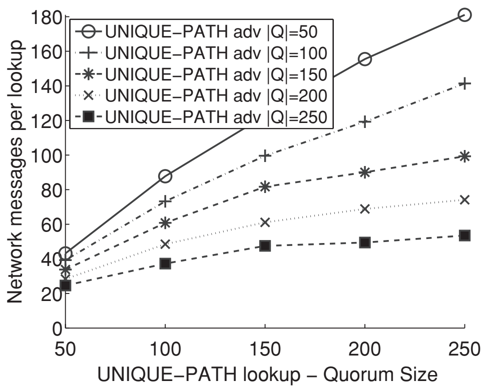
Fig. 11. RANDOM `advertise`, FLOODING `lookup`. $d_{\text{avg}} = 10$.

0.5 is achieved for TTL = 2, whereas a hit ratio of 0.85 is achieved for TTL = 3. The number of messages sent by FLOODING is quite small and is comparable with the PATH strategy. This is mainly due the broadcast nature of flooding: in the last hop of flooding, nodes that do not rebroadcast the flooding message further on can still reply to the `lookup` request if they posses the searched data.

However, notice that in order to increase the hit ratio to 0.9 in a network of 800 nodes one has to increase the TTL to 4, resulting in a significant communication cost increase. Instead of sending 14 messages with TTL 3, FLOODING with TTL 4 sends 35 messages (this is both due to the increased flooding scope and additional reply messages). This demonstrates the biggest disadvantage of the FLOODING strategy: coarse coverage granularity and the lack of a fine-grain control over the intersection probability (as explained in Section 4.4). This is in contrast with the PATH strategy, in which one can control the intersection probability in a very fine-grain manner by changing the number of nodes that need to be covered by the random walk.



(a) hit ratio. static networks



(b) number of messages per lookup. static networks

Fig. 12. UNIQUE-PATH `advertise`, UNIQUE-PATH `lookup`, $n = 800$, $d_{\text{avg}} = 10$.

In mobile networks, FLOODING performs quite similar to static networks. Surprisingly, it achieves a slightly higher hit ratio for the same TTL as in static networks, while sending more messages. This is due to the mobility model artifact: it is well known that in the random waypoint model nodes tend to concentrate at the center of the network [Bettstetter et al. 2003]. Thus, the average density increases and as a result, given the same TTL, more nodes are covered and more messages are being sent.

8.5 UNIQUE-PATH `advertise` with UNIQUE-PATH `lookup`

We have also explored the possibility to access the `advertise` quorum with the UNIQUE-PATH strategy. As proven in Theorem 5.5, at least one of the random walks has to be of length $\Omega(\frac{n}{\log n})$. Figure 12 depicts the hit ratio for various random walk TTL values for a network of 800 nodes. Both `advertise` and `lookup` were accessed by UNIQUE-PATH, rather than by PATH, which considerably improved their performance. We can see that a 0.9 hit ratio is achieved when the length of `advertise` random walk and `lookup` random walk together is around 340, almost $n/2$. Thus, if both walks use the same target quorum size, it must equal approximately $|Q_a| = |Q_\ell| = 170 \approx 1.5\frac{n}{\log n} \approx n/4.7$. For such a choice of quorum sizes, the number of messages of the `lookup` access is about $|Q_\ell|/2$, since the first hit occurs at approximately half the way. Note, however, that the exact constants of the crossing time depend on the network topology and density and are not the same for all networks. Thus, the target quorum sizes cannot be set in a generic way. This is in contrast with the RANDOM x UNIQUE-PATH mix, in which the same quorum sizes guarantee the same intersection on any topology and density.

8.6 Fast Mobility Impact

We now turn our attention to exploring the impact of faster mobility on the performance of probabilistic quorums. According to Figure 13(a), when applying the access strategies exactly as before, the hit ratio deteriorates as the maximal

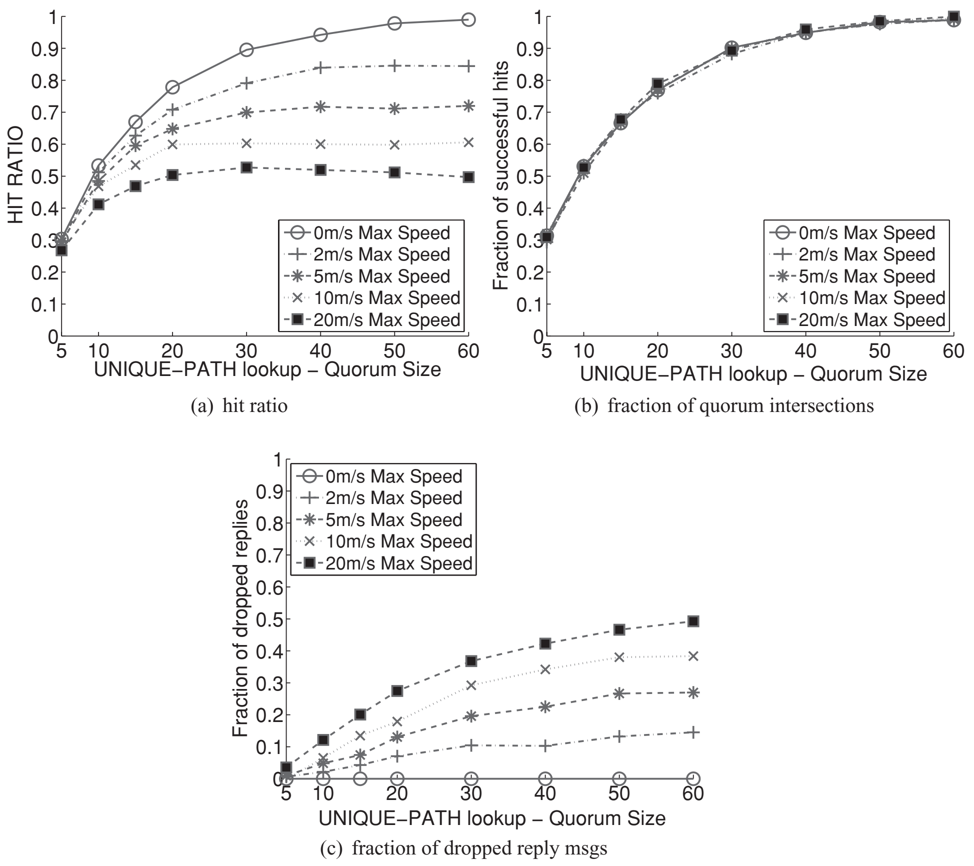
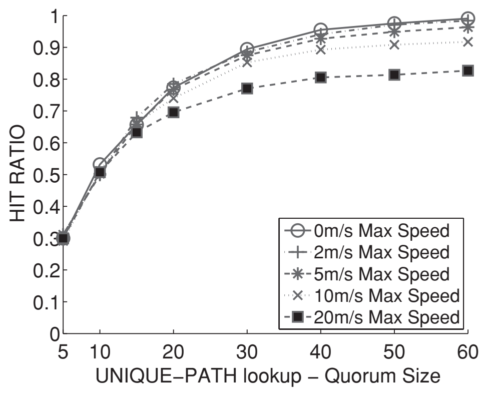


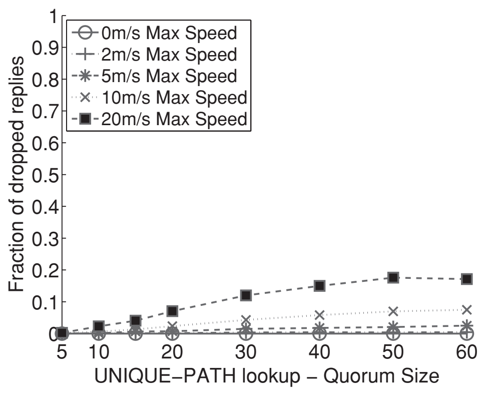
Fig. 13. RANDOM `advertise`, UNIQUE-PATH `lookup`, varying mobility, $n = 800$, without reply-path local repair. Due to high mobility the reply messages do not arrive back.

speed increases. To shed a light on the reasons for this effect, we have looked at every stage of our protocol. Recall that to access a `lookup` quorum by the UNIQUE-PATH strategy, the random walk first traverses the network until it encounterers $|Q_\ell|$ different nodes, stops upon the first hit, and then sends the reply back to the looking node, following the reverse random walk path. We can see in Figure 13(b) that the intersection probability itself (not taking into account the fail of the reply) is not impacted by mobility at all. This is because we use the *RW salvation* technique to prevent dropping of random walk messages, as discussed in Section 6.2. However, when following the reverse path, a growing number of reply messages are being dropped (Figure 13(c)). Basically, this happens since when a node is not able to reach a certain node that is the next hop along the reverse random walk path, it drops the reply message. Naturally, the longer the random walk is, the higher is the chance that at least one of the nodes that was previously traversed by the random walk will move out of the range and the reverse path will break.

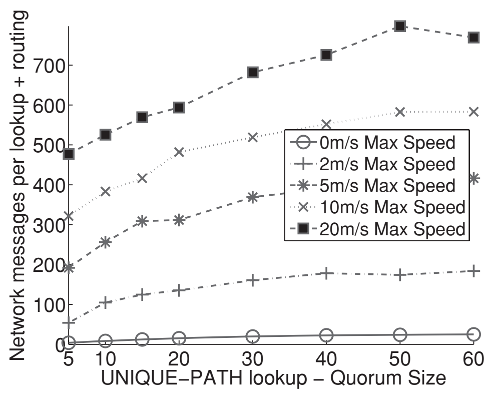
We have tackled fast mobility with the reply-path local repair technique, also discussed in Section 6.2. Figures 14(a) and 14(b) depict the hit ratio with



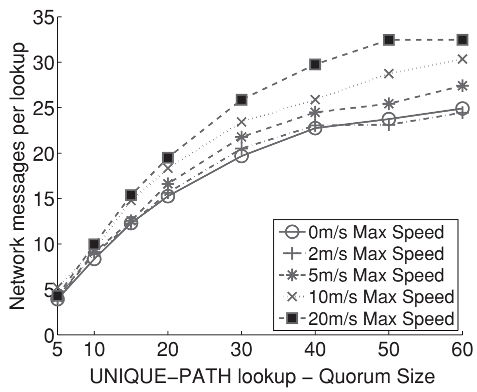
(a) hit ratio. advertise $|Q|=2\sqrt{n}$



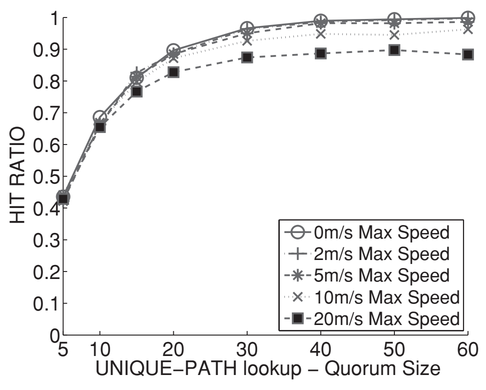
(b) fraction of dropped reply messages



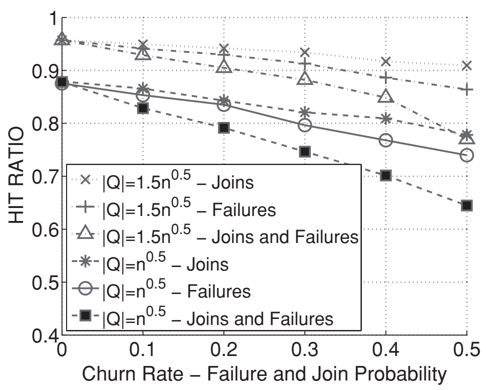
(c) num. msgs. per lookup + routing



(d) number of messages per lookup



(e) hit ratio. advertise $|Q|=3\sqrt{n}$



(f) degradation rate in face of churn. static. $d_{\text{avg}} = 15$

Fig. 14. RANDOM `advertise`, UNIQUE-PATH `lookup`, varying mobility, $n = 800$, with reply-path local repair. Reply-path local repair technique helps reply messages to arrive back, despite the high mobility.

reply-path repairs. We can see that the combination of local and global path repairs fixes the hit ratio and decreases the amount of reply message droppings significantly. As for the networking price, Figure 14(c) depicts the amount of network messages including routing control messages (AODV path establishment and maintainable). As speed increases, more and more reply-path repairs are required and the price of routing increases. Note, however, that if there was no need to send the reply back, the RW salvation technique alone fully eliminates the mobility effects almost without any cost. The random walk itself does not invoke routing at all, and the amount of generated network messages is almost the same for static and mobile networks (Figure 14(d)).

Another way to combat fast mobility in a proactive way is by increasing the `advertise` quorum. According to Figure 14(e), increasing $|Q_{\texttt{advertise}}|$ from $2\sqrt{n}$ to $3\sqrt{n}$ improves the hit ratio, since the `lookup` access needs shorter random walks, which also decreases the chance for a reply-path breakage. Another possible way to deal with reply-path breakages is by incorporating anycast logic into the routing layer, in a more tight cross-layer design. Exploring this option is left for future work.

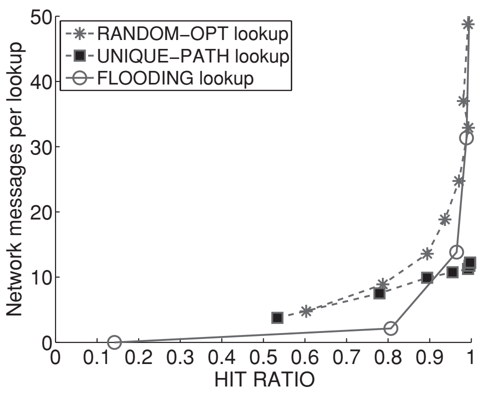
8.7 Churn

Figure 14(f) depicts the intersection probability in face of churn (in static 800 node network, with average density of 15 neighbors, which kept the network connected in all scenarios). After all advertisements finished, we fail every node with a given probability or/and add new nodes according to the average churn rate (the x-axis). The size of the `lookup` quorum was adjusted to the new network size. We can see an outstanding survivability of the probabilistic quorums: the intersection probability deteriorates very slowly with increasing churn rate. For example, an initial intersection probability of 0.95 degrades to only 0.87 in face of as much as 50% failures (as long as the network remains connected). This also matches our analysis from Section 6.1.

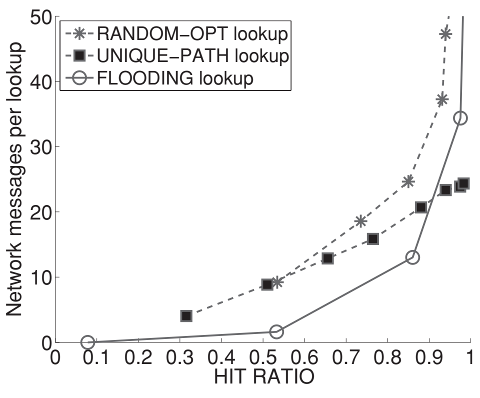
8.8 Simulation Summary

Figure 15 presents a comparison summary of 3 `lookup` strategies. We plot the number of messages per lookup request that are required to achieve the same intersection probability. The depicted values for RANDOM-OPT do not include the price of routing messages. Even without taking routing into account RANDOM-OPT is inferior to the other two strategies. For low intersection probability FLOODING outperforms UNIQUE-PATH. However, for larger intersection probability we need to increase the flooding TTL, which increases the number of messages disproportionately to the additional gained intersection. This demonstrates the lack of fine-grained control of FLOODING over the search scape.

Figure 16 presents a more detailed summary of our simulation study for some specific setup values. For `lookup` we have depicted both the cost of a hit and the cost of a miss. In case of a miss (the looked-up object is not present), the whole cost of a target quorum size is paid. In case of a hit, this cost is reduced due to the early halting of some strategies (but also includes the price of a



(a) $n = 200$, mobile network



(b) $n = 800$, mobile network

Fig. 15. Comparison of 3 `lookup` strategies: hit ratio vs. number of messages per lookup. RANDOM `advertise`, $d_{\text{avg}} = 10$.

Advertise		RANDOM				UNIQUE-PATH
Lookup		RANDOM	RANDOM-OPT	UNIQUE-PATH	FLOODING	UNIQUE-PATH
Advertise Cost	Static	600				250
	Mobile	1600				
Cost of Lookup Miss	Static	350*	40*	33	50**	100
	Mobile	1000*	100*			
Cost of Lookup Hit	Static	350*	40*	23	35	35
	Mobile	1000*	100*			

* this does not include the routing cost
** 50 messages is the coverage range of flooding with TTL 3. Smaller TTL results in a non -sufficient intersection

Fig. 16. Summary of simulation results for combinations of different access strategies, $n = 800$, $d_{\text{avg}} = 10$, intersection 0.9, $|Q_a| = 2\sqrt{n} = 56$, $|Q_\ell| = 1.15\sqrt{n} = 33$.

reply). RANDOM and RANDOM-OPT lookup quorums were accessed serially and not in parallel, thus do not benefit from early halting.

Based on these values and our result from Section 5.4, we can reason which strategy mix is better. For example, the relative cost of `advertise` versus `lookup` for a RANDOM $\otimes$ UNIQUE-PATH combination in this setting is $\frac{|Q_a|Cost_a}{|Q_\ell|Cost_\ell} = \frac{600}{33} = 18$, while for a UNIQUE-PATH $\otimes$ UNIQUE-PATH combination it is $\frac{|Q_a|Cost_a}{|Q_\ell|Cost_\ell} = \frac{250}{100} = 2.5$. Therefore, in this setting, as long as $\tau > 2.5$ (τ is the frequency of `lookups` versus `advertises`), it is more efficient to use RANDOM $\otimes$ UNIQUE-PATH rather than UNIQUE-PATH $\otimes$ UNIQUE-PATH.

9. RELATED WORK

9.1 Quorum Systems

Quorum systems were implicitly introduced by Gifford [1979] and Thomas [1979] as weighted voting mechanisms for ensuring consistency. An explicit definition of quorums later appeared in Garcia-Molina and Barbara [1985], and was extended to bicoteries, and therefore biquorums, in Mitchell et al. [1992]. Herlihy used quorums and consensus to implement shared objects in Herlihy [1986]. Other quorum-based implementations of distributed shared memory (or shared objects) include Attiya et al. [1995] and Lynch and Shvartsman [2002]. Probabilistic quorum systems were initially introduced by Malkhi et al. [2001]. They were later explored, for example, in Malkhi and Reiter [1998] and Miura and Tagawa [2006].

Reconfigurable quorum systems were first explored by Herlihy [1987]. Additional dynamical reconfiguration mechanisms of quorum systems appeared in Lynch and Shvartsman [2002] and Abraham and Malkhi [2003], yet without analyzing the failure probability of a single quorum. Moreover, a method for dynamic update of quorums in the face of joins and leaves based on maintaining de Bruijn graph was presented in Abraham and Malkhi [2003]. Those methods are unsuitable for ad hoc networks due to their large message complexity. Our calculations of the degradation rate can serve as a simple mechanism for determining how often a quorum system needs to be refreshed to ensure continued intersections (or lookup success).

Byzantine resilient quorum systems were investigated, for example, in Alvisi et al. [2000], Malkhi and Reiter [1998], and Martin and Alvisi [2004]. In order to deal with Byzantine systems, the size of the intersection of quorums must be large enough to mask possible false output by Byzantine nodes. In this article, we do not address Byzantine failures.

An implementation of a probabilistic quorum system for sensor networks appears in Chockler et al. [2006]. The access to both write and read quorums is performed by flooding (gossiping) a corresponding request throughout the entire network. In writes, the data is saved by $\Theta(n)$ nodes, yet only $\Theta(\log n)$ nodes send an acknowledgment. For reads, the ids of $\Theta(\log n)$ random nodes are included in the header of the corresponding message; only these nodes are supposed to send a reply to the read request. This mix of access strategies is a special case of `advertise` FLOODING, `lookup` RANDOM strategies, while the choice of random `lookup` requires a random membership or sampling procedure, just like in our work.

9.2 Quorum-Based Location Services

One of the most widely used applications of quorums in ad hoc networks thus far has been in implementing location services. In quorum-based protocols (also known as *rendezvous-based*), all nodes in the network agree, implicitly or explicitly, upon a mapping that associates each node's unique identifier to one or more other nodes in the network. The mapped-to nodes are the location servers for that node. They are the nodes where periodical location updates are being stored and location queries will be routed to and looked up at. The mapping of nodes to quorums should be such that the update quorum will intersect with the lookup quorum. Examples of quorum-based location services include, Haas and Liang [1999a], Stojmenovic [1999], and Hubaux et al. [2001], Octopus [Melamed et al. 2005], LLS [Abraham et al. 2004], and GLS [Li et al.

2000]. Additional examples of data location services are GCLP [Tchakarov and Vaidya 2004], GHT [Ratnasamy et al. 2002], and Rendezvous Regions [Seada and Helmy 2003]. All these works differ from ours in that most of them use geographic knowledge, do not use probabilistic quorums, and do not utilize asymmetric quorum systems. In that respect, our work is the first systematic study of probabilistic quorum systems in ad hoc networks.

In the works of Haas and Liang [1999a, 1999b], a uniform random quorum system is used for mobility management. Node location information is maintained in location databases that form a virtual backbone. When a node moves, it updates its location with one quorum containing the nearest backbone node. Each source node then queries the quorum containing its nearest backbone for the location of the destination, and uses that location to route the message. In Haas and Liang [1999b] the division of nodes into quorums is static and done a priori, while ensuring uniformity in the sense that all nodes will be members of the same number of quorum sets. On the other hand, in Haas and Liang [1999a] the selection of nodes into quorums is done randomly during runtime.

In PAN [Luo et al. 2004, 2003] both read and write quorums are random subsets of nodes. The write quorum is accessed by random gossip which also constructs random membership. The read quorum is accessed by contacting a set of random nodes picked out of the random membership directly (relying on routing), in the same way as in our RANDOM access. Both write and read quorums are directed only to a predefined subset of nodes, termed *Storage Set* (`StS`). StS can be any subset or overlay (such as connected dominating set), picked statically or dynamically and agreed upon by all nodes. However, PAN does not specify how to dynamically reconfigure the quorum system when StS changes. Our scheme could be extended in a straightforward way to use only a subset of nodes for quorums. Since the write updates in PAN are disseminated to the whole network (or the whole StS) and are eventually stored by all StS nodes, the size of the read quorum can be kept relatively small (it must still be more than one, to overcome node failures and message loss). Our work could be seen as a generalization of Luo et al. [2003], since it provides a number of alternative quorum access strategies, while Luo et al. [2003] only provides RANDOM `advertise x` RANDOM `lookup`. In addition, our scheme is more flexible since it does not disseminate the write updates to all (StS) nodes, but allows to adjust the sizes of the write/read quorums optimally with respect to the access pattern, as shown in Section 5.4.

The work in Karumanchi et al. [1999] focuses on the problem of efficiently utilizing quorum systems in a highly dynamic environment. Nodes are partitioned into fixed quorums, and every operation updates a randomly selected set, thus balancing the load. Their simulation study indicates that probabilistic quorums have a better recency rate than other strict quorum strategies.

In GeoQuorums [Dolev et al. 2003], geometric coordinates determine the location of home servers. These focal point coordinates define geographic areas that must be inhibited by at least one server at any time. Sets of focal points are organized in intersecting quorums. The quorums are further used to implement an atomic memory abstraction in mobile ad hoc networks. The algorithm to dynamically reconfigure the set of available quorums is presented as well. At this stage, we focus on implementation of quorum systems and location services without utilizing geographic information.

In Bhattacharya [2003], a location service is implemented through randomly selected quorums. Yet, no means are provided in Bhattacharya [2003] to determine the required size of the random quorum and no theoretical evaluation of the quorum selection algorithms is supplied.

10. DISCUSSION

In this article we have explored various access strategies for implementing probabilistic biquorum systems in ad hoc networks. In particular, we have shown that asymmetric biquorums can offer better performance than symmetric ones. Moreover, we have shown that even without geographical knowledge (or localization), it is possible to obtain efficient quorums. The biquorum system we have found most efficient is the one that uses RANDOM for `advertise` and UNIQUE-PATH for `lookup` (or vice versa). This is due to the use of random walks in UNIQUE-PATH which eliminates the need for multiple-hop routing.

We would like to stress that, as mentioned in the Introduction, asymmetric constructions of probabilistic biquorum systems are useful not only for ad hoc networks, but also for any network with nonuniform access costs (e.g., peer-to-peer networks). This is because in many workloads, one type of quorum access (e.g., lookup) is much more frequent than the other (e.g., advertise). Hence, the more common access type may communicate only with the closest nodes, and only the rare access type needs to communicate with random nodes, which are most certain to include some far away nodes. Of course, the result holds also when advertisements are more frequent than lookups, in which case advertisements are the ones that can be performed on nearby nodes.

The main driving application that we addressed in this work is data location services, which can also be trivially generalized to distributed dictionary services and bulletin boards. Yet another appealing application of quorums is distributed shared objects. In particular, it was shown in Attiya et al. [1995] that atomic registers, also known as linearizable read/write objects [Herlihy and Wing 1990], can be implemented using quorums. Yet such an implementation requires both read and write operations to access one `advertise` and one `lookup` quorum [Attiya and Welch 1998; Lamport 1986; Lynch 1996].[6] When using probabilistic quorums, these protocols in fact implement what is known as probabilistic linearizability [Gramoli 2007].

An interesting area for future research is the usage of probabilistic quorums for implementing decentralized publish/subsribe (pub/sub) systems [Baldoni et al. 2005]. For example, a natural way of implementing pub/sub using quorums is to disseminate a subscription to all members of an advertise quorum;

[6]There are certain optimizations by which when there is no contention, some of these accesses can be saved, but in the worst case, some read and some write operations have to access both quorums [Dutta et al. 2004].

a published event can be sent to all members of a lookup quorum. Each member of the lookup quorum checks if the event matches any of the subscriptions it is aware of, and if one exists, it sends a notification to the corresponding subscriber. In particular, as event publications typically occur much more frequently than subscriptions, the use of an asymmetric biquorum system like the ones we propose here is advantageous. Clearly, when using probabilistic quorums, the guarantees of the resulting pub/sub system become probabilistic as well. Notice, however, that an interesting challenge of such a system is how to execute unsubscriptions efficiently. That is, in probabilistic quorum systems each access to a quorum touches a possibly different set of nodes. Hence, sending an unsubscribe message to a single probabilistic quorum might not be enough.

APPENDIX

A. RANDOM WALK PRELIMINARIES

Let $G(V, E)$ be an undirected graph, with V the set of nodes and E the set of edges. Let $n = |V|$ and $m = |E|$. For $v \in V$, let $N(v) = \{u \in V \mid (v, u) \in E\}$ be the set of neighbors of v, and let $\delta(v) = |N(v)|$ be the degree of v.

Let $X_v = \{X_v(\tau) : \tau \geq 0\}$ be a simple random walk starting from node v on the state space V with transition matrix Q. When the walk is at node v, the probability to move in the next step to u is $Q_{vu} = \Pr(v, u) = \frac{1}{\delta(v)}$ for $(v, u) \in E$ and 0 otherwise. Let $X_v(t) = \{X_v(\tau) : t \geq \tau \geq 0\}$ denote a walk of length t. Let $\mathcal{N}(t) = |X_v(t)|$ denote the number of distinct nodes visited by the random walk of length t.

The *hitting time*, $h(u, v)$, is the expected time for a random walk starting at u to arrive to v for the first time. Let $h_{\max}$ be the maximum $h(u, v)$ over all ordered pairs of nodes. The return time $h(v, v)$ is the expected time to return to v for the first time, starting at v. $h(v, v)$ is well known to be equal to $\frac{1}{\pi_v}$, from the theory of Markov chains [Lovász 1993] where $\pi = \{\pi_v : v \in V\}$ is the stationary distribution of the simple random walk. In Avin and Ercal [2007] the authors prove the following.

PROPOSITION A.1. *For a constant $c > 1$, if $r^2 \geq \frac{c \cdot 8 \log n}{n}$, then with high probability for $\mathcal{G}^2(n, r)$:*

(i) *$h_{\max}$ is linear in n. That is, there exists a constant γ independent of n, such that, with high probability $h_{\max} \leq \gamma n$.*
(ii) *$\forall v \ \pi_v = \Theta(\frac{1}{n}) \geq \frac{1}{\beta n}$ for a constant β independent of n.*

The first statement is based on a resistance argument and the second is based on proving that with high probability the degree of every node is $\Theta(\log n)$.

B. PARTIAL COVER TIME OF RANDOM GEOMETRIC GRAPHS

The *Partial Cover Time*, $PCT_G(i)$, of a graph G is the expected time taken by a simple random walk on G to visit i distinct nodes in G. Formally, for $v \in V$, let $PCT_v(i)$ be the expected number of steps needed for the simple random walk starting at v to visit i distinct nodes in G, and the partial cover time of G is $PCT(i) = \max_v PCT_v(i)$. The *cover time*, C_G, of G is the expected time to visit all nodes in G, that is, $C_G = \max_v PCT_G(n)$.

THEOREM 4.1 (RESTATED). *Let $t(n) = o(n)$. For a constant $c > 1$, if $r^2 \geq \frac{c \cdot 8 \log n}{n}$, then, for large enough n, with high probability for $\mathcal{G}^2(n, r)$*

$$PCT_G(t) \leq 2\alpha t,$$

where α is a constant.

PROOF. The proof proceeds as follows. We will first bound the expected number of visits to each vertex during the walk and then show that the number of distinct nodes visited by the random walk of length $t = o(n)$ is at least $t/(2\alpha)$.

Given a graph G and a starting node v, we view the random walks as proceeding in phases. For $1 \leq i \leq n$ we identify phase i with a set of i vertices, $V_i \subseteq V$ and a starting vertex $s_i \in V$, which is the last vertex visited in phase $i-1$. Phase i starts with the random walk at s_i and ends with the random walk exiting V_i. Note that $V_1 = v$. If the walk is of length t and $k \leq t$ distinct nodes are visited during the walk, then for $k < j \leq n$, we define s_j and V_j to be empty sets.

For now consider only $X_v(t)$ where we may omit t if it is clear from the context. Let α_i denote the number of visits to s_i during the t steps. If $s_i = \emptyset$ then $\alpha_i = 0$. Clearly

$$t = \sum_1^n \alpha_i.$$

Taking expectation over all possible walks of length t we have

$$E[t] = t = E\left[\sum_1^n \alpha_i\right] = \sum_1^n E[\alpha_i].$$

Consider the graph in the theorem, we now show that the expected number of visits to each vertex during $X_v(t)$, is bounded by a constant.

LEMMA B.1. $\max_i E[\alpha_i] \leq \alpha$.

PROOF. The proof is based on the Proposition A.1. Let $t = o(n)$. For a random walk X_v, let $\mathcal{E}_v^t$ denote the event that the random walk returns to v within time t and $\bar{\mathcal{E}}_v^t$ the complementary event. Let $\rho_v = \Pr(\mathcal{E}_v^t)$. For the $\mathcal{G}^2(n, r)$ given in the theorem we can bound the return time in the following way.

$$\begin{aligned}
h(v, v) &= \frac{1}{\pi_v}, && (1)\\
&= \beta n, && (2)\\
&\leq \rho_v E[h(v, v) \mid \mathcal{E}_v^t] + (1 - \rho_v) E[h(v, v) \mid \bar{\mathcal{E}}_v^t], && (3)\\
&\leq \rho_v t + (1 - \rho_v)(t + h_{\max}), && (4)\\
&= t + (1 - \rho_v)(h_{\max}), && (5)\\
&\leq t + (1 - \rho_v)\gamma n, && (6)
\end{aligned}$$

where β, γ are the constants from Proposition A.1, which do not depend on n. Steps (2) and (6) are licensed by Proposition A.1. Step (4) is true since if the walk does not return to v within t steps we can bound the return time by taking t steps plus the time it takes to return to v from the node at which the walk is at time t. From the preceding we can conclude

$$\beta n \leq t + (1 - \rho_v)\gamma n. \tag{7}$$

Now, for large enough n, we can lower bound $(1 - \rho_v)$ as follows. We have

$$(1 - \rho_v) \geq \frac{\beta n - t}{\gamma n} \geq \frac{\beta n - o(n)}{\gamma n} \geq c_1, \tag{8}$$

where $0 < c_1 < 1$ is a constant not depending on n (e.g., c_1 could be $\frac{\beta}{2\gamma}$). Let R_v be the expected number of returns to v before step t. Using the bound on $1 - \rho_v$ (i.e., the probability the a walk starting at the v will not return to v in time t) we can bound R_v with the expected number of trials (every time the walk returns to v we start a new independent trial) until the walk does not return to v anymore in the t steps.

$$R_v \leq \frac{1}{1 - \rho_v} \leq 1/c_1 \tag{9}$$

By setting $\alpha = c_1 + 1$, we can conclude that the expected number of visits of a walk of length t to every s_i is at most α times (since we can see every phase i as a random walk of length less than t starting at s_i), so $\forall i \; E[\alpha_i] \leq \alpha$. □

Lemma B.2. *For a random walk $X_v(t)$, if $\forall i \leq n \;\; E[\alpha_i] \leq \alpha$, then*

$$E[\mathcal{N}(t)] \geq \frac{t}{2\alpha}. \tag{10}$$

Proof. We omit the t. We can express $E[\mathcal{N}]$ as follows.

$$E[\mathcal{N}] = \sum_{j=1}^{n} \Pr(\mathcal{N} \geq j) = \sum_{j=1}^{n} 1 - \Pr(\mathcal{N} < j) \tag{11}$$

Now we will bound $\Pr(\mathcal{N} < j)$ by $\Pr(\mathcal{N} < j) = \Pr(\mathcal{N} \leq j - 1) \leq (j - 1)\alpha/t$.

$$\Pr(\mathcal{N} \leq j) \leq \min\left\{\frac{j\alpha}{t}, 1\right\} \tag{12}$$

Clearly $\Pr(\mathcal{N} \leq j) \leq 1$. We will prove that $\Pr(\mathcal{N} \leq j) \leq j\alpha/t$. Let $\mathcal{E}_j$ be the event that $N \leq j$ and $\bar{\mathcal{E}}_j$ be its compliment. We can express $E[\alpha_i]$ via conditional expectation.

$$E[\alpha_i] = \Pr(\mathcal{E}_j)E[\alpha_i \mid \mathcal{E}_j] + \Pr(\bar{\mathcal{E}}_j)E[\alpha_i \mid \bar{\mathcal{E}}_j] \tag{13}$$

Assume by contradiction that $\Pr(\mathcal{E}_j) > \frac{j\alpha}{t}$. Since all walks are of length t, for all walks that visit at most j distinct nodes we have $t = \sum_{i=1}^{j} \alpha_i$, and

$$\sum_{i=1}^{j} E[\alpha_i \mid \mathcal{E}_j] = t.$$

Since the average value is t/j, there exist i^* for which $E[\alpha_{i^*} \mid \mathcal{E}_j] \geq t/j$. Otherwise, if $\forall i : E[\alpha_i \mid \mathcal{E}_j] < t/j$ we will get $\sum_{i=1}^{j} E[\alpha_i \mid \mathcal{E}_j] \leq j \cdot \max_j\{E[\alpha_i \mid \mathcal{E}_j]\} < j \cdot t/j = t$. Now, from Eq. (13) we get

$$E[\alpha_{i^*}] > \frac{j\alpha}{t} \cdot \frac{t}{j} + \Pr(\bar{\mathcal{E}_j})E[\alpha_{i^*} \mid \bar{\mathcal{E}_j}]. \tag{14}$$

$$> \alpha. \tag{15}$$

This is a contradiction to $\forall i \leq n\ E[\alpha_i] \leq \alpha$, so $\Pr(\mathcal{E}_j) \leq \frac{j\alpha}{t}$.

Let $k = \lfloor \frac{t}{\alpha} \rfloor$, now we can put everything together. We have

$$E[\mathcal{N}] = \sum_{j=1}^{n} 1 - \Pr(\mathcal{N} < j) \tag{16}$$

$$\geq \sum_{j=1}^{n} 1 - (j-1)\frac{\alpha}{t} \tag{17}$$

$$\geq \sum_{j=1}^{k} 1 - (j-1)\frac{1}{k} \tag{18}$$

$$\geq k - \frac{1}{k}\sum_{j=1}^{k-1} j \tag{19}$$

$$\geq k - \frac{1}{k}\frac{(k-1)k}{2} \tag{20}$$

$$= \frac{1}{2}(k+1) \tag{21}$$

$$> \frac{t}{2\alpha} \tag{22}$$

which proves the assertion of the Lemma. □

From Lemma B.2, the expected number of distinct nodes a walk visits in t steps is at least $t/2\alpha$. Thus, we conclude the statement of the theorem that $PCT_{\mathcal{G}}(t) \leq 2\alpha t$.

REFERENCES

Abraham, I., Dolev, D., and Malkhi, D. 2004. LLS: A locality aware location service for mobile ad hoc networks. In *Proceedings of the Joint Workshop on Foundations of Mobile Computing (DIALM-POMC)*. 75–84.

Abraham, I. and Malkhi, D. 2003. Probabilistic quorums for dynamic systems. In *Proceedings of the 16th International Symposium on Distributed Computing (DISC)*. 60–74.

Alvisi, L., Pierce, E. T., Malkhi, D., Reiter, M. K., and Wright, R. N. 2000. Dynamic Byzantine quorum systems. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN)*. 283.

Attiya, H., Bar-Noy, A., and Dolev, D. 1995. Sharing memory robustly in message passing systems. *J. ACM 42*, 1, 124–142.

Attiya, H. and Welch, J. 1998. *Distributed Computing: Fundamentals, Simulations and Advanced Topics*. McGraw Hill.

Avin, C. and Brito, C. 2004. Efficient and robust query processing in dynamic environments using random walk techniques. In *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks (IPSN)*. 277–286.

Avin, C. and Ercal, G. 2007. On the cover time and mixing time of random geometric graphs. *Theor. Comput. Sci. 380*, 1-2, 2–22.

Baldoni, R., Marchetti, C., Virgillito, A., and Vitenberg, R. 2005. Content-Based publish-subscribe over structured overlay networks. In *Proceedings of the 25th International Conference on Distributed Computing Systems (ICDCS)*. 437–446.

Bar-Yossef, Z., Friedman, R., and Kliot, G. 2008. RaWMS—Random walk based lightweight membership service for wireless ad hoc networks. *ACM Trans. Comput. Syst. 26*, 2, 1–66.

Barr, R., Haas, Z. J., and van Renesse, R. 2005. JiST/SWANS Java in simulation time/scalable wireless ad hoc network simulator. `http://jist.ece.cornell.edu/`.

Bettstetter, C., Resta, G. and Santi, P. 2003. The node distribution of the random waypoint mobility model for wireless ad hoc networks. *IEEE Trans. Mobile Comput. 2*, 3, 257–269.

Bhattacharya, S. 2003. Randomized location service in mobile ad hoc networks. In *Proceedings of the 6th ACM International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWIM)*. 66–73.

Broch, J., Maltz, D. A., Johnson, D. B., Hu, Y.-C., and Jetcheva, J. 1998. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the 4th ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*. 85–97.

Chockler, G., Demirbas, M., Gilbert, S., Newport, C., and Nolte, T. 2005. Consensus and collision detectors in wireless ad hoc networks. In *Proceedings of the 24th Symposium on the Principles of Distributed Computing (PODC)*.

Chockler, G., Gilbert, S., and Patt-Shamir, B. 2006. Communication-Efficient probabilistic quorum systems for sensor networks. In *Proceedings of the 4th International Conference on Pervasive Computing and Communication Workshops (PERCOMW)*. 111.

Chockler, G., Keidar, I., and Vitenberg, R. 2001. Group communication specifications: A comprehensive study. *ACM Comput. Surv. 33*, 4, 427–469.

Dolev, S., Gilbert, S., Lynch, N., Shvartsman, A., and Welch, J. 2003. Geoquorums: Implementing atomic memory in mobile ad hoc networks. In *Proceedings of the 17th International Symposium on Distributed Computing (DISC)*.

Dolev, S., Schiller, E., and Welch, J. 2002. Random walk for self-stabilizing group communication in ad hoc networks. In *Proceedings of the 21st ACM Symposium on Principles of Distributed Computing (PODC)*. 259–259.

Dutta, P., Guerraoui, R., Levy, R. R., and Chakraborty, A. 2004. How fast can a distributed atomic read be? In *Proceedings of the 23rd ACM Symposium on Principles of Distributed Computing (PODC)*. 236–245.

Feige, U. 1996. A fast randomized LOGSPACE algorithm for graph connectivity. *Theor. Comput. Sci. 169*, 2, 147–160.

Friedman, R. and Kliot, G. 2006. Location services in wireless ad hoc and hybrid networks: A survey. Tech. rep. CS-2006-10, Technion, Haifa, Israel. `http://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-info.cgi?2006/CS/CS-2006-10`.

Friedman, R., Raynal, M., and Travers, C. 2005. Two abstractions for implementing atomic objects in dynamic systems. In *Proceedings of the International Conference on Principles of Distributed Systems (OPODIS)*. 73–87.

Garcia-Molina, H. and Barbara, D. 1985. How to assign votes in a distributed system. *J. ACM 32*, 4, 841–860.

Gifford, D. K. 1979. Weighted voting for replicated data. In *Proceedings of the 7th Symposium on Operating System Principles (SOSP)*. 150–162.

Gkantsidis, C., Mihail, M., and Saberi, A. 2004. Random walks in peer-to-peer networks. In *Proceedings of the 23rd Conference of the IEEE Communications Society (InfoCom)*. 259–259.

Gramoli, V. 2007. Distributed shared memory for large-scale dynamic systems. Ph.D. thesis, University of Rennes 1.

Guerraoui, R. and Raynal, M. 2004. The information structure of indulgent consensus. *IEEE Trans. Comput. 53*, 4, 453–466.

GUPTA, P. AND KUMAR, P. 1998. Critical power for asymptotic connectivity in wireless networks. In *Stochastic Analysis, Control, Optimization and Applications*. 547–566.

GUPTA, P. AND KUMAR, P. 2000. The capacity of wireless networks. *IEEE Trans. Inf. Theory 46*, 2, 388–404.

HAAS, Z. AND LIANG, B. 1999a. Ad hoc mobility management with randomized database groups. In *Proceedings of IEEE International Conference on Communications (ICC)*.

HAAS, Z. AND LIANG, B. 1999b. Ad hoc mobility management with uniform quorum systems. *IEEE/ACM Trans. Netw. 7*, 2, 228–240.

HERLIHY, M. 1986. A quorum-consensus replication method for abstract data types. *ACM Trans. Comput. Syst. 4*, 1, 32–53.

HERLIHY, M. 1987. Dynamic quorum adjustment for partitioned data. *ACM Trans. Datab. Syst. 12*, 2, 170–194.

HERLIHY, M. AND WING, J. 1990. Linearizability: A correctness condition for concurrent objects. *ACM Trans. Program. Lang. Syst. 12*, 3, 463–492.

HOC NETWORKS WORKING GROUP, I. M. A. 2007. MANET neighborhood discovery protocol (NHDP). `http://tools.ietf.org/html/draft-ietf-manet-nhdp-07.txt`.

HUBAUX, J.-P., GROSS, T., BOUDEC, J.-Y. L., AND VETTERLI, M. 2001. Towards self-organizing mobile ad hoc networks: The terminodes project. *IEEE Comm. Mag. 39*, 1, 118–124.

IEEE COMPUTER SOCIETY. 2007. 802.11: Wireless LAN media access control (MAC) and physical layer (PHY) specifications. http://standards.ieee.org/getieee802/802.11.html.

IETF MOBILE AD-HOC NETWORKS WORKING GROUP. 2008. MANET neighborhood discovery protocol. http://tools.ietf.org/html/draft-ietf-manet-nhdp-07.txt.

IETF MOBILE AD-HOC NETWORKS WORKING GROUP. 2008. RFC 5148: Jitter considerations in mobile ad hoc networks (MANETs). `www.ietf.org/rfc/rfc5148.txt`.

KARUMANCHI, G., MURALIDHARAN, S., AND PRAKASH, R. 1999. Information dissemination in partitionable mobile ad hoc networks. In *Proceedings of the Symposium on Reliable Distributed Systems*. 4–13.

KEIDAR, I. AND MELAMED, R. 2006. Evaluating unstructured peer-to-peer lookup overlays. In *Proceedings of the ACM Symposium on Applied Computing (SAC)*. 675–679.

KESHAVARZ-HADDAD, A., RIBEIRO, V., AND RIEDI, R. 2006. Broadcast capacity in multihop wireless networks. In *Proceedings of the 12th International Conference on Mobile Computing and Networking (MobiCom)*. 239–250.

KLIOT, G. 2008. Wireless signal interference models made simple. `www.cs.technion.ac.il/~gabik/Jist-Swans/signal_interference`.

LAMPORT, L. 1986. On interprocess communication. *Distrib. Comput. 1*, 2, 77–101.

LI, J., JANNOTTI, J., DE COUTO, D., KARGER, D., AND MORRIS, R. 2000. A scalable location service for geographic ad-hoc routing. In *Proceedings of the 6th International Conference on Mobile Computing and Networking (MobiCom)*. 120–130.

LOVÁSZ, L. 1993. Random walks on graphs: A survey. *Combinatorics 2*, 1–46.

LUO, J., EUGSTER, P., AND HUBAUX, J. 2004. Pilot: Probabilistic lightweight group communication system for ad hoc networks. *IEEE Trans. Mobile Comput. 3*, 2, 164–179.

LUO, J., HUBAUX, J.-P., AND EUGSTER, P. 2003. PAN: Providing reliable storage in mobile ad hoc networks with probabilistic quorum systems. In *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*. 1–12.

LV, C., CAO, P., COHEN, E., LI, K., AND SHENKER, S. 2002. Search and replication in unstructured peer-to-peer networks. In *Proceedings of the 16th International Conference on Supercomputing (ICS)*. 84–95.

LYNCH, N. 1996. *Distributed Algorithms*. Morgan Kaufman.

LYNCH, N. A. AND SHVARTSMAN, A. A. 2002. RAMBO: A reconfigurable atomic memory service for dynamic networks. In *Proceedings of the 16th International Conference on Distributed Computing (DISC)*. 173–190.

MADRAS, N. AND SLADE, G. 1993. *The Self-Avoiding Walk*. Birkhauser, Boston, MA.

MALKHI, D., REITER, M., WOOL, A., AND WRIGHT, R. 2001. Probabilistic quorum systems. *The Inform. Comput. J. 170*, 2, 184–206.

MALKHI, D. AND REITER, M. K. 1998. Secure and scalable replication in phalanx. In *Proceedings of the 17th IEEE Symposium on Reliable Distributed Systems (SRDS)*. 51.

MARTIN, J.-P. AND ALVISI, L. 2004. A framework for dynamic byzantine storage. In *Proceedings of the 34th International Conference on Dependable Systems and Networks (DSN)*. 325.

MASSOULIE, L., MERRER, E. L., KERMARREC, A.-M., AND GANESH, A. J. 2007. Peer counting and sampling in overlay networks: Random walk methods. *Distrib. Comput. 20*, 4, 267–278.

MELAMED, R., KEIDAR, I., AND BAREL, Y. 2005. Octopus: A fault-tolerant and efficient ad-hoc routing protocol. In *Proceedings of the 24th IEEE Symposium on Reliable Distributed Systems (SRDS)*. 39–49.

MITCHELL, N., MIZUNO, M., AND RAYNAL, M. 1992. A general method to define quorums. In *Proceedings of the 12th International Conference on Distributed Computing Systems (ICDCS)*. 657–664.

MIURA, K. AND TAGAWA, T. 2006. A quorum-based protocol for searching objects in peer-to-peer networks. *IEEE Trans. Parall. Distrib. Syst. 17*, 1, 25–37.

MOTWANI, R. AND RAGHAVAN, P. 1995. *Randomized Algorithms*. Cambridge University Press.

PANCHAPAKESAN, P. AND MANJUNATH, D. 2001. On the transmission range in dense ad hoc radio networks. In *Proceedings of IEEE Signal Processing Communication (SPCOM)*.

PENROSE, M. D. 2003. *Random Geometric Graphs*. Oxford University Press.

RATNASAMY, S., KARP, B., YIN, L., YU, F., ESTRIN, D., GOVINDAN, R., AND SHENKER, S. 2002. GHT: A geographic hash table for data-centric storage in sensornets. In *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*.

ROWSTRON, A. AND DRUSCHEL, P. 2001. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility. In *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP'01)*. 188 – 201.

SALTZER, J. H., REED, D. P., AND CLARK, D. D. 1984. End-to-End arguments in system design. *ACM Trans. Comput. Syst. 2*, 4, 277–288.

SEADA, K. AND HELMY, A. 2003. Rendezvous regions: A scalable architecture for service provisioning in large-scale mobile ad hoc networks. In *Proceedings of ACM SIGCOMM*. (Refereed poster.)

SERVETTO, S. AND BARRENECHEA, G. 2002. Constrained random walks on random graphs: Routing algorithms for large scale wireless sensor networks. In *Proceedings of the ACM International Workshop on Wireless Sensor Networks and Application (WSNA)*.

STOJMENOVIC, I. 1999. A routing strategy and quorum based location update scheme for ad hoc wireless networks. Computer Science, SITE, University of Ottawa, TR-99-09.

TCHAKAROV, J. AND VAIDYA, N. 2004. Efficient content location in wireless ad hoc networks. In *Proceedings of the IEEE International Conference on Mobile Data Management (MDM)*.

THOMAS, R. H. 1979. A majority consensus approach to concurrency control for multiple copy databases. *ACM Trans. Datab. Syst. 4*, 2, 180–209.

TOH, C. 2002. *Ad Hoc Mobile Wireless Networks*. Prentice Hall.

Received February 2009; revised June 2010; accepted July 2010