Probabilistic Quorum Systems in Wireless Ad Hoc Networks

Roy Friedman Gabriel Kliot
Computer Science Department
Technion - Israel Institute of Technology
Haifa 32000, Israel
{roy,gabik}@cs.technion.ac.il

Chen Avin
Communication Systems Engineering Dep.
Ben Gurion University of The Negev
Beer Sheva, Israel
avin@cse.bgu.ac.il

Abstract

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 paper explores several access strategies for implementing probabilistic quorums in ad hoc networks. In particular, we present the first detailed study of asymmetric probabilistic bi-quorum systems and show its advantages in ad hoc networks. The paper includes both a formal analysis of these approaches backed by a simulation based study. In particular we show that one of the strategies, based on Random Walks, exhibits the smallest communication overhead, thus being very attractive for ad hoc networks.

1 Introduction

Quorums are a basic construction in any distributed systems. They can be used as building blocks (or at least as design pattern) in solving various fundamental problems such as Consensus [18], distributed dictionaries and location services [14], distributed storage [3, 29], 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 [31].

The idea behind quorums is that they ensure intersection between subsets of nodes. 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 completing operations which 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, 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 [14], a very large percentage of location services are based on bi-quorums. Recently, there have been several attempts to solve Consensus and distributed storage in ad hoc networks, e.g., [8, 11, 27]. As mentioned before, these also require quorums.

In this paper we investigate the implementation of scalable probabilistic quorums in ad hoc networks. 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 network (caused by churn and nodes mobility) makes the usage of strict deterministic quorums highly costly. Hence, for the sake of scale and efficiency, we relax the requirements of the quorum system to probabilistic ones, similar to [31].

One could potentially use geographical knowledge for construction of quorum systems in ad hoc network (e.g., [24, 40]). 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 paper we look for quorum systems that do not rely on geographical knowledge.

Contributions of this work. We propose several schemes for accessing probabilistic quorums in ad hoc networks. We study the performance of the proposed schemes both analytically and by simulations. In particular, one of the schemes we investigate is based on random walks (RW). The use of RWs in wireless ad hoc networks has been previously proposed to solve various problems such as membership construction [12, 6], reliable multicast [12], routing [38] and querying [4]. RWs are attractive for ad hoc networks since they require neither multi-hop routing nor broadcasting, which are expensive in ad hoc networks [6]. Also, they offer fine grain control over the communication overhead as well as early halting capabilities, as elaborated later in this paper.

An important contribution of this work is the introduction of the first *asymmetric* probabilistic bi-quorum system. Specifically, in previous works, e.g., [26, 27, 31], accesses to all quorums of a probabilistic bi-quorum system are performed using the same access strategy.[1] In that respect, all previously known probabilistic bi-quorum systems are *symmetric*. In our work, we show that it is possible to combine different access strategies (and different quorum sizes) and still obtain intersection with high probability. Moreover, in

[1] In [26, 27], the authors use the term "asymmetric probabilistic quorums", however the asymmetry in [26, 27] refers to different quorum sizes and not different access strategies.

ad hoc networks, such asymmetric bi-quorum systems offer superior performance compared to symmetric ones.

Additionally, for random walks, we provide a definition of the *crossing time* of RWs and a lower bound on the crossing time in random geometric graphs. We also investigate the *partial cover time (PCT)* of RWs in random geometric graphs when small fractions of the network should be covered. To capture the reliability of the quorum systems in dynamic environments we present a new metric, called *degradation rate*.

Our simulations are carried on the JIST/SWANS simulator from Cornell, which models mobility, collisions, and signal propagation, and implements the entire 802.11 MAC.

2 Preliminaries

Quorums and Bi-Quorums. Intuitively, a quorum system is a set of subsets such that every two subsets intersect. Moreover, a bi-quorum system consists of two sets of subsets such that each subset in one set intersects with each subset in the other set. Below, we provide a formal definition of these notions, following the works of [15, 17, 21, 35].

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 bi-quorum 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 bi-quorums. We will also refer to them here as `lookup` and `advertise` quorums given that bi-quorums are often used in conjunction with lookup and advertise operations.[2] However, the discussion applies the same for any bi-quorum 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.

Probabilistic Quorums. In *probabilistic quorums* [31], 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 bi-quorums, such as lookup/advertise quorums, it is ensured that each (randomly selected) `lookup` quorum intersects with every (randomly selected) `advertise` quorum with a given probability.

Ad Hoc Network System Model. Consider a set of nodes spread across a geographical area and communicating by exchanging messages using a wireless medium. Each node v may send messages that can be received by all other nodes within its transmission range r_v. A node u is a *neighbor* of another node v if u is located within the transmission range of v[3]. The combination of the nodes and the neighborhood relations forms a wireless ad hoc network.

The network connectivity graph of an ad hoc network is is often modelled as a two dimensional *Random Geometric Graph*(RGG) [37], denoted $G^2(n,r)$. Each node knows all of its direct neighbors (by using a simple heartbeat mechanism that is present in any case in most routing algorithms for ad hoc networks) and can communicate with them directly. A node can also communicate with other distant nodes whose address it knows by applying routing. Nodes do not know their position and we do not use any geographic knowledge.

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.

3 Quorum Systems Metrics

Any implementation of a probabilistic quorum system can be analyzed according to the following quality measures [31]:
Intersection probability: Probabilistic quorum system $\mathcal{Q}$ is an ε-intersecting if the total access probability of pairs of intersecting quorums is at least $1-\varepsilon$.
Access cost: The cost (in messages) of accessing a quorum.
Load: The request load on a single node. The target is to balance the request load equally between 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 [31], the fault tolerance of a probabilistic quorum system of size $\ell\sqrt{n}$ is $n - \ell\sqrt{n} + 1 = \Omega(n)$. In ad hoc network it is also required that the quorum nodes form a connected graph.[4]

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 [31], the failure probability of a probabilistic quorum system of size $\ell\sqrt{n}$ is $e^{-\Omega(n)}$ for all $p \leq \ell - \frac{l}{\sqrt{n}}$. Again, in ad hoc network, quorum nodes must also form a connected graph.

Failure resilience refers to the resilience of the whole quorums system to failures. As long as the entire quorum system has not failed, a live quorum can be found. But it does not capture the chance of survival of a previously accessed quorum. In this work we introduce the following novel measure:

[2]We discuss implementing read/write registers via quorums in Sec 9.

[3]In practice, the transmission range does not behave exactly as a disk due to various physical phenomena. However, this does not affect the access strategies, but greatly simplifies the formal model. At any event, our simulation results are carried on a simulator that simulates a real transmission range behavior including distortions, background noise, unidirectional links, etc.

[4]We elaborate on connectivity of ad hoc networks in Section 5.2.

Advertise	RANDOM				FLOODING		PATH
Lookup	RANDOM	RANDOM-OPT*	PATH	FLOODING	PATH	FLOODING	PATH
Advertise Cost	$\frac{n}{\sqrt{\ln(n)}}$				Combined Cost n	Combined Cost n	$\frac{n}{\ln(n)}$ ***
Lookup Cost	$\frac{n}{\sqrt{\ln(n)}}$	$\sqrt{n\ln(n)}$	$\sqrt{n}$	$\sqrt{n}$			$\frac{n}{\ln(n)}$ ***
Lookup Routing	Yes	Yes	No	No	No	No	No
# Replies	Multiple**	Multiple**	One	Multiple	One	Multiple	One
Early Halting	No**	No**	Yes	No	Yes	No	Yes

* RANDOM with the cross layer optimization, discussed in section 6
** unless accessed serially
*** this is a lower bound, also validated by simulations

Figure 1. Asymptotic and qualitative comparison of different access strategies for $|Q| = \Theta(\sqrt{n})$.

Degradation rate: The probability of a given quorum to stay alive as a function of time when individual nodes crash independently with fixed probability. For probabilistic quorums, this translates to the probability that two quorums accessed at different times will intersect despite the fact that between these two accesses f nodes have crashed. Hence, degradation rate captures the resilience of a single quorum in the face of dynamic changes. The degradation rate helps determining when should the quorum system be reconfigured, or refreshed, in order to recover from failures and node departures.

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 above. In a bi-quorum system, it is possible to mix and match between the access strategies used for lookup quorums and those used for advertise quorums based, e.g., on the relative frequency of requests of each type.

In this paper 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 underlying network graph for a predefined number of steps (time to live - TTL) to cover a sufficient set of different nodes. *FLOODING* performs a limited scope flooding of the network, which covers a set of different nodes. We also consider several optimizations for these basic techniques. The main novelty of our approach is an ability to mix those strategies in different ways, achieving various tradeoffs discussed below. More specifically, we show that in order to construct probabilistic quorum systems, not every quorum has to be accessed with a *RANDOM* strategy, as was previously done, e.g., in [27, 31]. Instead, some quorums can be accessed by other, more attractive methods.

Figure 1 provides a summary of the asymptotic costs and qualitative properties of various combinations of access strategies. In this table, we explore combinations of all strategies for implementing `advertise` and `lookup` quorums (note that the results are analogous if we switch strategies between `lookup` and `advertise`). *RANDOM-OPT* stands for the *RANDOM* strategy with the cross layer optimization, discussed in Section 6. In all strategies, a quorum of size $\Theta(\sqrt{n})$ is accessed (except for *PATH*x*PATH*, *FLOODING*x*FLOODING* and *FLOODING*x*PATH*). We elaborate on each of the strategies and the corresponding entries in the table below. Yet, even before delving into the details, it can already be seen that the *PATH* strategy used for accessing `lookup` quorums wins in all categories: it has the lowest cost, does not require additional routing, has an early stopping property and does not incur a burden of sending multiple replies. We now turn to the details:

***RANDOM* access strategy.** In this method, a quorum (be it `lookup` or `advertise`) is simply any random, uniformly chosen, set of nodes Q. In order to choose Q, we can either rely on a membership service or sample the nodes directly.

Membership Service based Implementation. If a list of all node ids in the system is available (this list can be obtained through a standard membership service [10]), we can simply randomly select node ids from this set. Alternatively, we can utilize a random membership service for ad hoc networks, such as RaWMS [6], which provides the required sampling capabilities. Once the ids of the nodes 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 multi-hop routing, which includes both the cost of using the routes and establishing the routes. It is well known ([19]) that the diameter of the unit disk 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 $\Theta(|Q| \cdot 1/r) = O(|Q|\sqrt{\frac{n}{\ln n}})$. When $|Q| = \Theta(\sqrt{n})$, the price 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 routes reuse. For 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 starting an

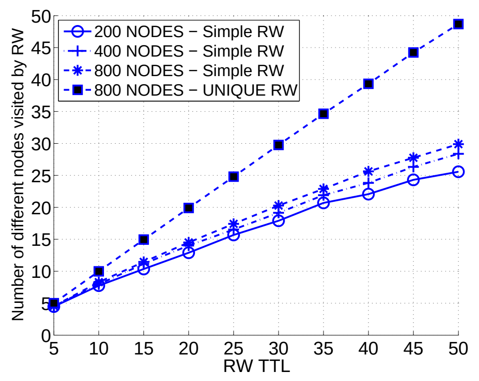


Figure 2. Unique nodes visited by RW.

appropriate number of Maximum Degree RWs (MD RW) [6] every time a quorum should be accessed. Every node is uniformly sampled from the network by a single MD RW, whose length equals the network's mixing time. 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 size of k, we may need more than k RWs. However, for similar arguments as in the birthday paradox, as long as the quorum size is at most $O(\sqrt{n})$, the chance of such collisions are very small. Hence, no more than $O(\sqrt{n})$ RWs are needed (for a more precise analysis refer to [6]). Using the analysis of [6] of the mixing time of a MD RW in ad hoc networks, each of these RWs should be started with TTL=$n/2$. 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.

***PATH* access strategy.** Another way to pick a quorum is by performing a single RW, which traverses the underlying network graph for a sufficient number of steps to cover $|Q|$ different nodes. Naturally, this number should be set such as to guarantee the intersection property of the quorum system.

The number of steps that it takes for a simple RW to visit i different nodes is called the *partial cover time* of i nodes and is denoted by $PCT(i)$ (for a given graph). It is already known that for Random Geometric Graphs $PCT(n/2) \leq O(n)$ [4] and the full cover time is $PCT(n) = O(n \log n)$ [5]. However, $PCT(i)$ for $i < n/2$ for $G^2(n,r)$ is generally unknown. Since Random Geometric Graphs are highly symmetric, we conjecture that the rate of discovering new nodes during the random walk is only decreasing over time. Based on this and $PCT(n/2) \leq O(n)$, we conclude that covering $\sqrt{n}$ nodes is linear in $\sqrt{n}$. We formulate the following conjecture:

Conjecture 4.1 *For $c > 1$, if $r^2 \geq \frac{c8\ln(n)}{n}$, then w.h.p. for $G^2(n,r)$ $PCT(\ell\sqrt{n}) = \Theta(\sqrt{n})$, for constant ℓ.*

We have validated Conjecture 4.1 empirically. Figure 2 depicts simulation results for the number of unique nodes visited by a single RW (and *UNIQUE-PATH* RW explained below), for different network and quorum sizes. In these runs, performed on the same setting as reported in Section 7, we run a RW with a given TTL and count the number of different visited nodes (averaging across multiple RWs and runs). For example, with $n = 800$, in order to visit $\sqrt{800} = 28$ nodes, a simple random has a length of 45, thus $PCT(\sqrt{n}) = 1.6\sqrt{n}$.

When implementing lookup operations *PATH* quorums have the following advantage: whenever the searched data has indeed been published, the RW is likely to find the advertisement before the TTL expires. In this case, called *early halting*, a reply can be returned immediately and the RW stopped. This reduces the communication overhead of lookups. As we show by simulations, it usually halves the length of the RW.

***UNIQUE-PATH* access strategy.** An optimization of the *PATH* strategy is to perform the RW in a way that prevents the RW from visiting nodes more than once, also known as *self-avoiding* RW [30]. This can be implemented, e.g., by storing the list of all nodes visited by the RW in its message header. In a rare event that all the neighbors of a current node have been visited by that RW, a random neighbors is chosen (as in simple RW). This optimization is expected to increase the efficiency of the *PATH* strategy since more different nodes are visited for the same number of steps at the cost of increased communication bit-complexity. As can be seen from Figure 2, empirically, indeed *UNIQUE-PATH* almost never revisits a node (at least for quorums of size $O(\sqrt{n})$).

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 either *PATH* or *UNIQUE-PATH* is being used, it would be beneficial to send the reply back on the reverse path of the RW (thus eliminating the need to invoke costly routing to send the reply). In such case, one needs to store the ids of all nodes visited by the RW in the message header anyway. Thus, if we want to send a reply back on the reverse path, *PATH* has no benefits over *UNIQUE-PATH* and the latter should be used.

***FLOODING* access strategy.** Another way to access a quorum is by a limited scope flooding. That is, the request is broadcasted from a given node to all its neighbors with a given 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 picking to take part in the `advertise` quorum with probability $|Q|/n$.

The main challenge in using *FLOODING* is how to set the TTL in order to ensure that the message is received by k nodes, for a given k. Of course, TTL can be always overestimated, resulting in accessing more than the minimal required number of nodes. However, this comes at the increased communication cost. Notice that for quorum intersection properties, k is often a function of n (e.g., $k = 2\sqrt{n}$). Whenever the density of the network is uniform and known, the TTL can be approximated. However, due to the reliance on broadcast (and especially in dynamic networks, in which nodes move and the network density changes), it is hard to have a fine grain control on the exact number of nodes that receive the message.

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 TTL expires. One can use the *expended ring* strategy to overcome this problem, however this also comes at an increased price. Another disadvantage for location services is the numerous number of replies that will be sent back to the looking node, which also increases the communication cost. Last but not least, broadcasting in wireless networks is less reliable and energy efficient than sending point-to-point messages.

5 Properties of Probabilistic Quorum Systems

As mentioned before, we can use any of the access strategies mentioned in Section 4 to implement any of the `advertise` and `lookup` quorums and we can mix and match them. Below, we present a formal evaluation of quorum systems obtained by several of these combinations.

5.1 Intersection Probability and Cost

`advertise` *RANDOM*, `lookup` *RANDOM*. This is the method of Malkhi et al. [31]. Lemma 3.4 from [31] states:

Lemma 5.1 *Let Q_a and Q_b be quorums of size $\ell\sqrt{n}$ each chosen uniformly at random. Then the non-intersection probability is $Pr(Q_a \cap Q_b = \varnothing) < e^{-\ell^2}$.*

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 below in Lemma 5.2) that if Q_a and Q_b are quorums of sizes $|Q_a|$ and $|Q_b|$ accordingly, each chosen uniformly at random, then $Pr(Q_a \cap Q_b = \varnothing) < e^{-\frac{|Q_a||Q_b|}{n}}$.

As we have shown above in Section 4, for $|Q| = \Theta(\sqrt{n})$ the cost of *RANDOM* access in ad hoc network is $\frac{n}{\sqrt{\ln n}}$ plus the cost of establishing multi-hop routes in case random membership is used or $\Theta(n\sqrt{n})$ when sampling directly with RWs.

`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 along the path of a RW with a given TTL.

Lemma 5.2 *Denote Q_a an `advertise` RANDOM quorum and Q_b a `lookup` PATH quorum. The probability that $Q_a \cap Q_b$ is empty is: $\Pr(Q_a \cap Q_b = \varnothing) \leq e^{-\frac{|Q_a||Q_b|}{n}}$.*

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_b was selected first.

Thus, suppose that a subgroup Q_b 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_b and Q_a to be empty, we can look at this process as if the first node of Q_a is picked out of $n - |Q_b|$ nodes uniformly at random, the next node is picked out of $n - |Q_b| - 1$ nodes, etc. Hence, we have:

$$\Pr(Q_a \cap Q_b = \varnothing) = \prod_{i=0}^{|Q_a|} \frac{n - |Q_b| - i}{n - i} \leq \prod^{|Q_a|} \frac{n - |Q_b|}{n} =$$

$$(1 - \frac{|Q_b|}{n})^{|Q_a|} \leq e^{-\frac{|Q_a||Q_b|}{n}}$$

■

The above result can be used to set the size of the quorums Q_a and Q_b to guarantee a non-empty intersection of two quorums with a given probability. This is established below:

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

Proof:

$$\Pr(Q_a \cap Q_b \neq \varnothing) \geq 1 - e^{-\frac{|Q_a||Q_b|}{n}} \geq 1 - \varepsilon \Rightarrow \ln(\varepsilon) \leq -\frac{|Q_a||Q_b|}{n}$$

■

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

Such 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 (non adversarial) way, has a significant advantage over the pure *RANDOM* strategy mix. While the cost of a *RANDOM* access is almost linear in n (plus a cost of establishing multi-hop routes), the cost of a *PATH* access is linear in the quorum size for $|Q| = \Theta(\sqrt{n})$.

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

Definition 5.1 *Given two RWs starting at any two nodes in the network, the* crossing time *is the expected first time at which there is a node that was visited by the two RWs.*

Formally, given two random walks X_u, Y_v, starting at u and v respectively, let $\gamma_{uv} = \min\{t : \{X_u(0), \ldots, X_u(t)\} \cap \{Y_v(0), \ldots, Y_v(t)\} \neq \emptyset\ \}$, 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 5.4 *The crossing time of two RWs in $G^2(n,r)$ is $\Omega(r^{-2})$.*

Proof: Divide the unit square of $G^2(n,r)$ into bins of size $\frac{1}{r} \times \frac{1}{r}$. 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 $\Omega(r^{-2})$ [25]. ■

Note that when $r = \Theta(\sqrt{\frac{\log n}{n}})$ (the minimal radius that ensures connectivity), the crossing time of $G^2(n, r)$ is at least $\Omega(\frac{n}{\log n})$. Therefore, if both `advertise` and `lookup` quorums are accessed by *PATH* strategy, at least one of these accesses will incur communication cost which is almost linear in n. Recall that this is a lower bound, and, as a matter of fact, our simulation results in Section 7 indicate that **both** `advertise` and `lookup` need to be of that order. In addition, there is an increased memory cost. The `advertise` RW has to publish the data item in every node it visits and if the length of this RW is $\Omega(n/\log n)$, accessing the `advertise` quorum will incur almost linear memory.

***UNIQUE-PATH* based strategies.** In each of the schemes using *PATH*, we can replace it with *UNIQUE-PATH*. This does not change the intersection probability, but can lower the communication cost. We provide a detailed simulation study of the *UNIQUE-PATH* strategies in Section 7.

***FLOODING* based strategies.** We can replace the *PATH* strategy used for `lookup` with *FLOODING* in each respective scheme. The intersection probability in these cases is similar to the ones obtained with *PATH* `lookup`. This is because a *FLOODING* that covers (at least) k nodes, visits k specific nodes, just like a specific RW. However, due to the broadcast nature of *FLOODING*, is 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.

5.2 Failures and Dynamism

Fault Resilience and Connectivity. Fault tolerance and failure probability were already discussed in Section 3. Here, we elaborate on the necessary connectivity condition of ad hoc networks. According to Gupta and Kumar [19] $G^2(n, r)$ is connected w.h.p if the transmission range r is set such that $r \geq \sqrt{\frac{C \ln n}{\pi n}}$ for $C > 1$. Such r implies an average number of neighbors, d_{avg}, which is $\pi r^2 n = C \ln n$.

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, $G^2(n - i, r)$. 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 network.

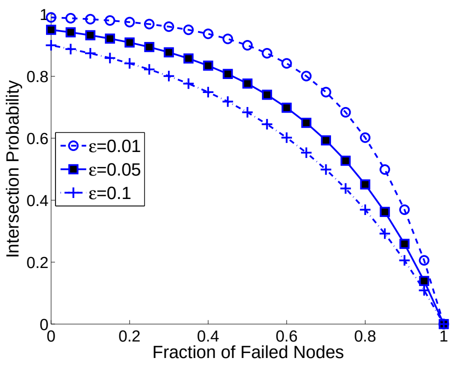


Figure 3. The degradation of the intersection probability as a function of the f - the fraction of crashed nodes. ε is the initial non-intersection probability.

Degradation Rate. We analyze the degradation rate as a function of the percentage of crashed nodes, assuming nodes crash independently. We calculate the probability of an intersection of a `lookup` quorum with a previously established `advertise` quorum. Denote by $Q_a(f)$ the live nodes of a given `advertise` quorum after a fraction of f nodes have failed. $Q_a(0)$ is the initial `advertise` quorum (before any node crashes) that guarantees intersection with at least $1 - \varepsilon$ probability. $|Q_a(f)| = |Q_a(0)| - f|Q_a(0)|$. `lookup` quorums access only live nodes at the moment the access is being issued so they are not affected by failures. Thus, the intersection probability after a fraction f of nodes fails is:

$$\Pr(Q_a(f) \cap Q_b(0) \neq \varnothing) \geq 1 - e^{-\frac{|Q_a(f)||Q_b(0)|}{n}} \geq$$
$$1 - e^{-\frac{|Q_a(0)|(1-f)|Q_b(0)|}{n}} = 1 - (e^{-\frac{|Q_a(0)||Q_b(0)|}{n}})^{1-f} \geq 1 - \varepsilon^{1-f}$$

Figure 3 illustrates the evolution of the intersection probability as a function of the number of failures. It can be seen that a bi-quorum system can withstand failures of a linear fraction of nodes. When starting with an initial intersection probability of 0.95, after a failure of 30% of the nodes, the intersection probability deteriorates to only slightly above 0.9.

Handling Dynamism. There are two sources of dynamism in ad hoc networks. The first is churn, caused by nodes arrivals, departures, and failures, while the second is mobility. The main problem that can be caused by churn is that due to nodes' failures and departures, the intersection probability with old quorum accesses degrades, as analyzed above.

Unlike deterministic quorum systems, in 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, e.g., by re-advertising every data item to ensure data continuity. The frequency of this re-advertising is determined by the degradation rate. Consider an example in Figure 3. Suppose the minimum accepted intersection probability of a given system is 0.9, the intersection probability when there are no failures is 0.95, and the time it takes 30% of the nodes to crash is one day. Then in this example, every data item should be refreshed once a day.

If mobility maintains the uniformly random distribution of nodes, then it does not impact the intersection probability.

However, if the mobility substantially skews the structure of the network, then refreshing by re-advertising is needed. The rate of refreshing depends on the exact mobility model.

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. Methods for estimating the size of an ad hoc network were discussed, e.g., in [6].

6 Optimizations

Service Dependent Optimizations - Caching. In the data location services the mapping of object to its home node remains valid for a long time. Thus, an additional caching of advertisement requests or a lookup replies that pass through nodes can significantly reduce the lookup overhead. With this optimization, and especially when using the *PATH* strategy, which has an early halting property, lookup requests for popular data items can terminate much faster and also have a higher chance of success.

Cross Layer Optimizations - *RANDOM-OPT* `lookup` Access Strategy. Since *RANDOM* utilizes routing, whenever a message passes through an intermediate node p, the networking layer of this node can invoke the data location service to perform a local lookup. If the data is found at p, p can respond immediately to the originator and instruct its own networking layer not to forward the lookup request any further. We denote this optimized access strategy by *RANDOM-OPT*. The benefit of this approach is that on average, an intersection can be achieved by contacting much fewer nodes. Specifically, if the advertisement used a *RANDOM* `advertise` quorum of size $\sqrt{n}$, then looking up in any set of $\sqrt{n}$ different nodes is enough to ensure intersection. Since the average length of a random route in the network is $\sqrt{\frac{n}{\ln n}}$, we can issue much fewer lookup requests. However, those routes may not necessarily pass in different nodes. As we show by simulation in Section 7, 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.

Random Walks Optimizations. We propose two additional optimizations for RW based techniques. The first one is called *path reduction*. It is applicable for *PATH* or *UNIQUE-PATH* `lookup` quorums and is used for reply messages. Whenever a lookup RW hits an `advertise` quorum, a reply message is sent over the reverse path of the RW. 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 appears 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, e.g., by switching their MAC to a promiscuous mode. If a node u that hears a RW `lookup` request passing through one of its neighbors v is part of the matching `advertise` quorum, u can send a reply immediately to v, which will stop the RW and send the reply back to the `lookup` originator. Thus, the number of nodes covered by a RW is significantly increased. Exploring the benefits of this technique is left for future work.

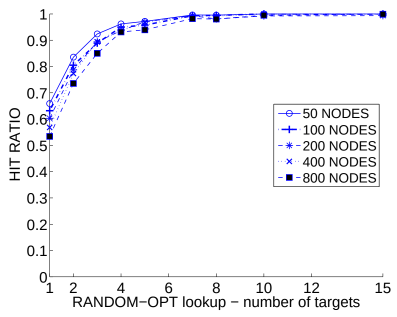
7 Simulations

Setup. The simulations were performed using the JiST-/SWANS simulator from Cornell university. We use the two-ray ground reflection model as the radio propagation model with IEEE 802.11 MAC protocol and 11Mb/sec throughput. The multi-hop routing protocol used for accessing *RANDOM* quorum is AODV. The mobility pattern was the Random Waypoint model with the speed of movement ranging from 0.5-2 m/s, which corresponds to slow and fast walking speeds, and an average pause time of 30s. All simulations were performed on networks of 50, 100, 200, 400 and 800 nodes.

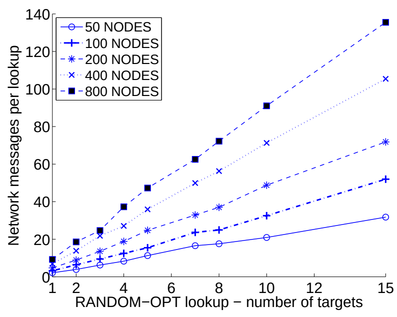
The nodes were placed at uniformly random locations in a square area. The transmission range r was fixed for all n at 220m. The average number of nodes in the transmission range of any node, d_{avg}, was set to 10. This was achieved by scaling the area size according to $a^2 = \frac{\pi r^2 n}{d_{\text{avg}}}$ and resulted in all networks being connected (according to the connectivity result of [19], d_{avg} should be $\pi r^2 n = C \ln n$, for $C > 1$ and in our case $d_{\text{avg}} = 10$ bounded $C \ln n$ for all n's we used).

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 seconds 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.

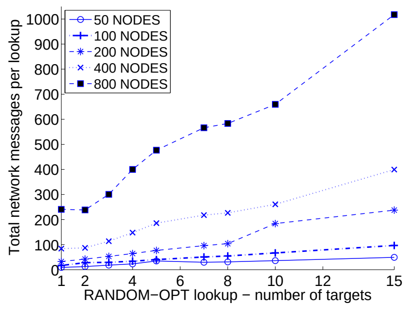
Simulation scenario. Each simulation comprised of two parts. In the first part a total number of 100 advertisements were performed by random nodes, each `advertise` by *RANDOM* access to a quorum of size $2\sqrt{n}$ (except for *UNIQUE-PATH* `advertise`). In the second part 1000 lookups were performed (by 25 random nodes, each making 40 lookups). `lookup` quorum was accessed by 4 different methods: *RANDOM*, *RANDOM-OPT*, *UNIQUE-PATH* and *FLOODING*. In case of a hit, a node sends a reply message to the node that originated the lookup request. In case of *RANDOM* and *RANDOM-OPT* the reply was sent by utilizing 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. Since we need to remember the RW path in order to send the reply back, there is no reason to use *PATH* (instead of *UNIQUE-PATH*). 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 stands for routing layer specific messages, which include 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. Due to the lack of space we did not include the study of additional `advertise` strategies, as well as of the degradation rate.



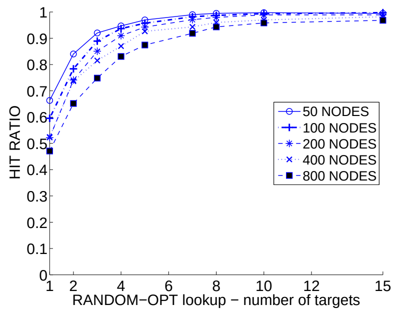
(a) Hit ratio, Static network



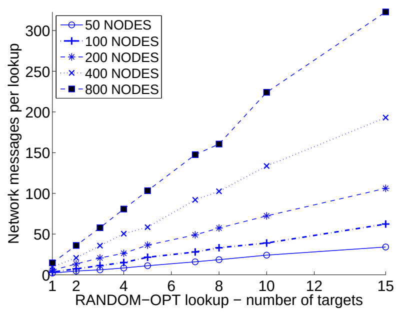
(b) Number of messages per lookup, Static network



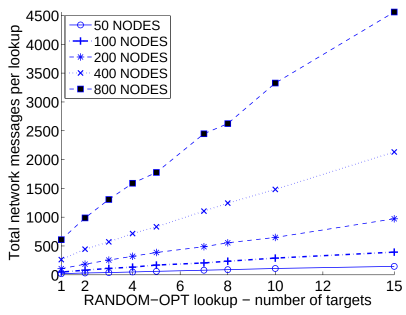
(c) Number of messages per lookup + routing, Static network



(d) Hit ratio, Mobile network



(e) Number of messages per lookup, Mobile network



(f) Number of messages per lookup + routing, Mobile network

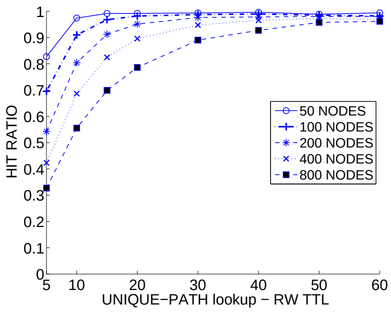
Figure 4. ***RANDOM* `advertise`, *RANDOM-OPT* `lookup`, Static and Mobile networks**

***RANDOM* `advertise` with *RANDOM* `lookup` and *RANDOM-OPT* `lookup`.** Due to the lack of space and since *RANDOM-OPT* performs better than *RANDOM*, we have included only the *RANDOM-OPT* figures. Still, we would like to mention that the behavior of *RANDOM* closely followed the theoretical analysis. For example, a hit ratio of 0.9 was achieved for a quorum size of $\approx 1.3\sqrt{n}$, as predicted by the analysis in Lemma 5.1. The number of messages per `lookup` request behaved as $\frac{|Q|\sqrt{n}}{\ln n}$. However, routing increases the communication overhead dramatically. This is primarily due to new routes establishment and route maintenance of AODV. Note that the price of establishing the routes is amortized over different quorum accesses due to routes reuse and its relative part drops in a longer run. However, in moving networks, when routes break and need to be reestablished, the price of routing stays a dominant performance factor.

Figure 4 depicts the performance of *RANDOM-OPT* `lookup` access strategy. 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, even if this message is not destined to this node, the actual accessed quorum size is $X\sqrt{\frac{n}{\ln n}} \approx \sqrt{n}$. 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 exactly $1.3\sqrt{n}$. The routing price of *RANDOM-OPT* is also much less than with *RANDOM*, since it uses fewer multi-hop routes. However, the additional cost of routing is still high, which makes this method very inefficient compared to the *UNIQUE-PATH* and *FLOODING* strategies.

In mobile networks the hit ratio of *RANDOM-OPT* `lookup` is only slightly smaller than the hit ratio achieved in static networks for the same quorum size. This is since about 10% of the messages are lost due to mobility and never reach their destination, mainly influencing the replies. In addition, the number of messages increases compared to static networks for the same quorum size. Generally speaking, the average path length in mobile network 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 destination. The routing price in mobile networks also dramatically increases.

***RANDOM* `advertise` with *UNIQUE-PATH* `lookup`.** Figure 5 depicts the performance of the *UNIQUE-PATH* access strategy in mobile networks. *UNIQUE-PATH* performed identically in mobile and static networks and thus we depict only the mobile case. A hit ratio of 0.9 is achieved when the target quorum size (RW TTL) is $\sim 1.3\sqrt{n}$, thus validating our analysis in Lemma 5.2 and testifying that a non random 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 sends. Surprisingly, accessing a target quorum of size $|Q|$ requires fewer than $|Q|$ messages, including the reply message! This


(a) Hit ratio, Mobile network

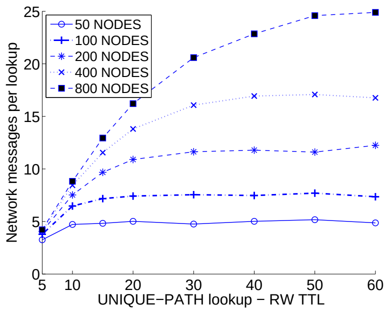
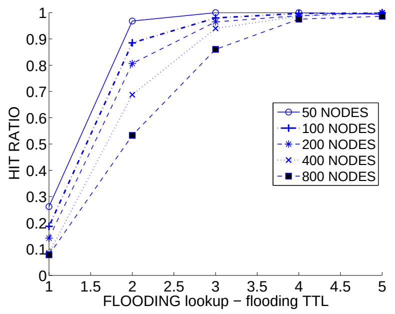
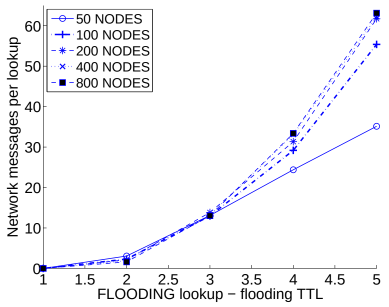

(b) Number of messages per lookup, Mobile network

Figure 5. *RANDOM* `advertise`, *UNIQUE-PATH* `lookup`, **Mobile networks**


(a) Hit ratio: *RANDOM* `advertise`, *FLOODING* `lookup`, Static network


(b) Number of messages: *RANDOM* `advertise`, *FLOODING* `lookup`, Static network

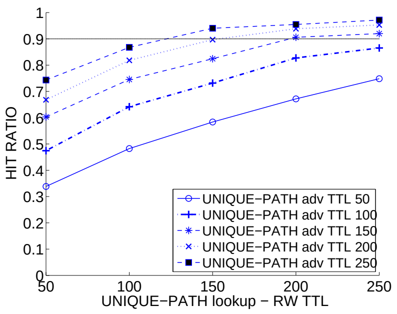

(c) Hit ratio: *UNIQUE-PATH* `advertise`, *UNIQUE-PATH* `lookup`, $n = 800$, Static

Figure 6. *RANDOM* **x** *FLOODING* **and** *UNIQUE-PATH* **x** *UNIQUE-PATH*, **Static networks**

happens due to the early halting. When a quorum of target size $|Q|$ is being accessed, the first hit occurs at approximately half of the way. Thus, an average of $|Q|/2$ messages are sent until hit. The reply message follows the reverse path, but due to the *path reduction* optimization described earlier, the reply path length is usually shorter. In addition, the node that originates the lookup always includes itself in the `lookup` quorum and checks if it can satisfy the request locally, which further reduces the number of messages by one.

An interesting implementation detail is the usage of the *salvation technique* to prevent dropping of RW messages. If node v does not succeed to forward a RW message to the neighbor chosen in a given step (did not receive a MAC level acknowledgement), 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.

***RANDOM* `advertise` with *FLOODING* `lookup`.** Figures 6(a) and 6(b) depict the performance of the *FLOODING* access strategy. The hit ratio grows super linearly with TTL. For example, for 800 nodes, a hit ratio of 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 to the *PATH* strategy. 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: the lack of a fine grained control over the intersection probability. This is in contrast to the *PATH* strategy. In mobile networks, *FLOODING* performs quite similar to static networks and is thus not depicted.

***UNIQUE-PATH* `advertise` with *UNIQUE-PATH* `lookup`.** As proven in Theorem 5.4, in this combination, at least one of the RWs has to be of length $\Omega(\frac{n}{\log n})$. Figure 6(c) depicts the hit ratio for various RW 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` RW and `lookup` RW together is around 350, almost $n/2$. Thus, if both walks use the same TTL, it must equal approximately $175 \approx 1.5\frac{n}{\log n} \approx n/5$. For such a choice of TTLs, the number of messages of the `lookup` access is about TTL/2, since the first hit occurs at approximately half the way.

8 Related Work

Quorum systems. Quorum systems were implicitly introduced by Gifford [16]. An explicit definition of quorums later appeared in [15], and was extended to bi-coteries, and therefore bi-quorums, in [35]. Herlihy used quorums and consensus to implement shared objects in [21]. Other quorum based implementations of distributed shared memory (or shared objects) include [3, 29]. Probabilistic quorum systems were initially introduced by Malkhi et al [31]. They were later explored, e.g., in [32, 36].

Reconfigurable quorum systems were first explored by Herlihy in [22]. Additional dynamical reconfiguration mechanisms of quorum systems appeared in [29] and [1], yet without analyzing the failure probability of a single quorum. Both methods are unsuitable for ad hoc networks due to their large message complexity.

Byzantine resilient quorum systems were investigated, e.g., in [2, 32, 33]. 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 paper, we do not address Byzantine failures.

An implementation of probabilistic quorum systems for sensor networks appears in [9]. By their method, the access to both write and read quorums is performed by flooding (gossiping) a corresponding request throughout the entire network. In the case of a write, the data is saved by $\Theta(n)$ nodes, yet only $\Theta(\log n)$ nodes send an acknowledgement. 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.

Quorum Based Location Services. One of the most widely used application of quorums in ad hoc networks thus far has been in implementing location service. Examples of quorum-based location services include, [39], Octopus [34], GLS [24], and GCLP [40]. All those works differ from our in that most of them use geographic knowledge, do not use probabilistic quorums and do not utilize a-symmetric quorum systems.

In the work of Haas and Liang [20], a uniform random quorum system is used for mobility management. Nodes 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. The selection of nodes into quorums is done randomly during runtime.

In [26, 27] both read and write quorums are random subsets of nodes. The write quorum is accessed by random gossip, which is also used to construct random membership. The read (query) 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. These works provide a detailed study of their quorum system, including an evaluation of the impact of nodes' failures on the reliability, in a similar way as our degradation rate. Our work could be seen as a generalization of the work in [27], since it provides a number of alternative quorum access strategies, while [27] only provides the *RANDOM* `advertise` x *RANDOM* `lookup` strategy.

In GeoQuorums [11], 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 atomic memory abstraction in mobile ad hoc networks.

In [7], a location service is implemented through randomly selected quorums. Yet, no means are provided in [7] to determine the required size of the random quorum and no theoretical evaluation of the quorum selection algorithms is supplied.

9 Discussion

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

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 [3] that atomic registers, also known as linearizable read/write objects [23], can be implemented using quorums. Yet, an implementation of such objects requires both read and write operations to access one `advertise` and one `lookup` quorum [28].[5] When using probabilistic quorums, these protocols in fact implement what is known as probabilistic linearizability [17].

References

[1] I. Abraham and D. Malkhi. Probabilistic quorums for dynamic systems. In *DISC*, pages 60–74, 2003.

[2] L. Alvisi, E. T. Pierce, D. Malkhi, M. K. Reiter, and R. N. Wright. Dynamic Byzantine Quorum Systems. In *DSN*, page 283, 2000.

[3] H. Attiya, A. Bar-Noy, and D. Dolev. Sharing Memory Robustly in Message Passing Systems. *J. ACM*, 42(1):124–142, 1995.

[4] C. Avin and C. Brito. Efficient and robust query processing in dynamic environments using RW techniques. In *IPSN*, pages 277–286, 2004.

[5] C. Avin and G. Ercal. On the cover time and mixing time of random geometric graphs. *Theor. Comput. Sci.*, 380(1-2), 2007.

[6] Z. Bar-Yossef, R. Friedman, and G. Kliot. RaWMS - Random Walk based Lightweight Membership Service for Wireless Ad Hoc Networks. In *MobiHoc*, pages 238–249, June 2006.

[7] S. Bhattacharya. Randomized location service in mobile ad hoc networks. In *MSWIM*, pages 66–73, 2003.

[8] G. Chockler, M. Demirbas, S. Gilbert, C. Newport, and T. Nolte. Consensus and Collision Detectors in Wireless Ad Hoc Networks. In *PODC*, 2005.

[9] G. Chockler, S. Gilbert, and B. Patt-Shamir. Communication-Efficient Probabilistic Quorum Systems for Sensor Networks. In *PERCOMW*, page 111, 2006.

[5]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 [13].

[10] G. Chockler, I. Keidar, and R. Vitenberg. Group Communication Specifications: a Comprehensive Study. *ACM Computing Surveys*, 33(4):427–469, 2001.

[11] S. Dolev, S. Gilbert, N. Lynch, A. Shvartsman, and J. Welch. Geoquorums: Implementing atomic memory in mobile ad hoc networks. In *DISC*, 2003.

[12] S. Dolev, E. Schiller, and J. Welch. Random Walk for Self-Stabilizing Group Communication in Ad Hoc Networks. In *PODC*, pages 259–259, 2002.

[13] P. Dutta, R. Guerraoui, R. R. Levy, and A. Chakraborty. How fast can a distributed atomic read be? In *PODC*, pages 236–245, 2004.

[14] R. Friedman and G. Kliot. Location Services in Wireless Ad Hoc and Hybrid Networks: A Survey. TR CS-2006-10, Technion, January 2006.

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

[16] D. K. Gifford. Weighted Voting for Replicated Data. In *SOSP*, pages 150–162, December 1979.

[17] V. Gramoli. *Distributed Shared Memory for Large-Scale Dynamic Systems*. PhD thesis, 2007.

[18] R. Guerraoui and M. Raynal. The Information Structure of Indulgent Consensus. *IEEE Trans. on Comput.*, 53(4):453–466, 2004.

[19] P. Gupta and P. Kumar. Critical Power for Asymptotic Connectivity in Wireless Networks. In *Stochastic Analysis, Control, Optimization and Applications*, pages 547–566, 1998.

[20] Z. Haas and B. Liang. Ad Hoc mobility management with randomized database groups. In *ICC*, June 1999.

[21] M. Herlihy. A quorum-consensus replication method for abstract data types. *ACM Trans. Comput. Syst.*, 4(1):32–53, 1986.

[22] M. Herlihy. Dynamic quorum adjustment for partitioned data. *ACM Trans. Database Syst.*, 12(2):170–194, 1987.

[23] M. Herlihy and J. Wing. Linearizability: a correctness condition for concurrent objects. *Trans. Prog. Lang. & Syst.*, 12(3):463–492, 1990.

[24] J. Li, J. Jannotti, D. De Couto, D. Karger, and R. Morris. A scalable location service for geographic ad-hoc routing. In *MobiCom*, pages 120–130, August 2000.

[25] L. Lovász. Random Walks on Graphs: A Survey. *Combinatorics*, 2:1–46, 1993.

[26] J. Luo, P. Eugster, and J. Hubaux. Pilot: Probabilistic lightweight group communication system for ad hoc networks. *IEEE Transactions on Mobile Computing*, 3(2):164–179, April 2004.

[27] J. Luo, J.-P. Hubaux, and P.Th. Eugster. PAN: Providing reliable storage in mobile ad hoc networks with probabilistic quorum systems. In *MobiHoc*, pages 1–12, 2003.

[28] N. Lynch. *Distributed Algorithms*. Morgan Kaufman, 1996.

[29] Nancy A. Lynch and Alexander A. Shvartsman. RAMBO: A Reconfigurable Atomic Memory Service for Dynamic Networks. In *DISC*, pages 173–190, 2002.

[30] N. Madras and G. Slade. *The self-avoiding walk*. Birkhauser, 1993.

[31] D. Malkhi, M. Reiter, A. Wool, and R. Wright. Probabilistic Quorum Systems. *The Information and Computation Journal*, 170(2):184–206, November 2001.

[32] D. Malkhi and M. K. Reiter. Secure and scalable replication in phalanx. In *SRDS*, page 51, 1998.

[33] J.-P. Martin and L. Alvisi. A framework for dynamic byzantine storage. In *DSN*, page 325, 2004.

[34] R. Melamed, I. Keidar, and Y. Barel. Octopus: A Fault-Tolerant and Efficient Ad-hoc Routing Protocol. In *SRDS*, October 2005.

[35] N. Mitchell, M. Mizuno, and M. Raynal. A General Method to Define Quorums. In *ICDCS*, pages 657–664, 1992.

[36] K. Miura and T. Tagawa. A quorum-based protocol for searching objects in peer-to-peer networks. *IEEE Trans. Parallel Distr. Syst.*, 17(1):25–37, 2006.

[37] M. D. Penrose. *Random Geometric Graphs*. Oxford Univ., 2003.

[38] S. Servetto and G. Barrenechea. Constrained Random Walks on Random Graphs: Routing Algorithms for Large Scale Wireless Sensor Networks. In *WSNA*, 2002.

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

[40] J. Tchakarov and N.H. Vaidya. Efficient Content Location in Wireless Ad Hoc Networks. In *MDM*, January 2004.