Open access • Proceedings Article • DOI:10.1109/CISS.2006.286425

Maximizing Path Durations in Mobile Ad-Hoc Networks — Source link

Yunghsiang S. Han, Richard J. La

Institutions: University of Maryland, College Park

Published on: 22 Mar 2006 - Conference on Information Sciences and Systems

Topics: Fast path, Any-angle path planning, Mobile ad hoc network, Path (graph theory) and Wireless ad hoc network

Maximizing Path Durations in Mobile Ad-Hoc Networks

Yijie Han and Richard J. La
University of Maryland, College Park
{hyijie,hyongla}@eng.umd.edu

***Abstract*— We study the problem of path selection in mobile ad-hoc networks. Based on our earlier results we propose a novel scheme that (i) maximizes the expected durations of selected paths and (ii) performs local path recovery by computing the probability a cached alternative path is still available in the event of a failure of a primary path in use. We evaluate the performance of our proposed scheme using ns-2 simulation and show that it yields a significant improvement in path durations.**

I. Introduction

The topology of a mobile ad-hoc network (MANET) is expected to be time-varying as the connectivity between nodes changes with time due to nodes' mobility. In order to cope with potentially frequent link failures and topology changes, researchers proposed many routing protocols for MANETs. Most of these protocols can be classified as either a table driven protocol or an on-demand routing protocol. Table driven protocols are proactive in that a node attempts to maintain a path to every known destination, whereas on-demand routing protocols are reactive and provide a path only when one is requested.

When one or more links along a path in use become unavailable (which we call a path failure), the path is no longer valid and a path recovery procedure is initiated to find an alternative path. In practice, detecting and recovering from a path failure can take a non-negligible amount of time (from applications' viewpoint), during which network service is disrupted. Frequent disruptions in network service will degrade the performance of applications, in particular time-critical applications. Moreover, an initiation of path recovery incurs additional overhead. Thus, in order to improve reliability of network service to applications and to minimize routing overhead, a good routing algorithm should take into consideration the duration of a selected path in addition to other qualities of the path, e.g., estimated available bandwidth. The duration of a path refers to the amount of time the path stays available till one of the links along the path goes down.

Predicting the duration of a path is difficult in practice as it depends on many factors that may not be known at the time of path selection. Most of existing routing protocols select a path based on some heuristic argument: The Dynamic Source Routing (DSR) protocol [5] selects the minimum hop path, whereas the Ad-hoc On-demand Distance Vector (AODV) routing protocol [7] selects the first discovered path. Associativity Based Routing (ABR) protocol [6] selects the path with maximum average age of the links. However, it is not clear how the hop count or the average age of the links along a path is related to its (expected) duration.

In order to this answer this question we studied the distribution of path duration and its relation to those of the links that provide them [2], [3]. Our main results show that (i) when the hop count of a path is large, the distribution of path duration can be approximated by an exponential distribution, and (ii) the parameter of the exponential distribution is determined by the expected durations of the links in the path.

In this paper, based on the results in [2], [3] we propose a novel scheme that (i) maximizes the expected durations of selected paths and (ii) enables local path recovery in the event of a path failure by switching to an alternative path cached at a node, whose estimated probability of being available exceeds some threshold. The latter exploits the well known memoryless property of an exponential distribution, and the threshold value can be adjusted to carry out a trade off between routing overhead and potentially larger delays during path recovery. Although our implementation does not consider other qualities of selected paths, it can easily be modified to take them into account. We evaluate the performance of the proposed scheme using ns-2 simulation. Simulation results suggest that our scheme significantly improves the statistical behavior of path durations.

The rest of the paper is organized as follows: We summarize our earlier work on the distribution of path duration in Section II. We describe our proposed scheme in Section III and how we can update the threshold for local path recovery to ensure that a selected alternative path is available with some target probability in Section IV. A detailed description of the implementation of our scheme in AODV and simulation results are provided in Section V.

II. A summary of earlier results

In this section we briefly summarize our earlier results reported in [2], [3] on the asymptotic distribution of path duration. We refer interested readers to [2], [3] for more details.

Consider a mobile ad-hoc network created and maintained by a set $V = \{1, \ldots, N\}$ of mobile nodes moving across a domain $\mathbb{D}$, which is a subset of $\mathbb{R}^2$ or $\mathbb{R}^3$, according to some mobility model. Links between two nodes are set up and torn down over time due to nodes' mobility; a link between them is established as soon as they become reachable, for example, when they move within a transmission range of each other and

packets from each other can be successfully decoded. When they become no longer reachable, the link is torn down.

In order to model the reachability between nodes, we introduce a $\{0,1\}$-valued *reachability* process $\{\xi_{ij}(t),\, t \geq 0\}$ for each pair of nodes $i, j \in V$. We assume that the link (i,j) from node i to node j is up (resp. down) at time $t \geq 0$ if $\xi_{ij}(t) = 1$ (resp. $\xi_{ij}(t) = 0$). The communication links are assumed bidirectional, whence we have $\xi_{ij}(t) = \xi_{ji}(t)$. The process $\{\xi_{ij}(t),\ t \geq 0\}$ is simply an alternating on-off process, with successive up and down time durations given by the random variables (rvs) $\{U_{ij}(k), k = 1, 2, \ldots\}$ and $\{D_{ij}(k), k = 1, 2, \ldots\}$, respectively.

Since the system is expected to run for a long time, we assume that the steady state has been reached and that the reachability processes are jointly stationary. This assumption is captured by requiring that the sequence of rvs $\{\boldsymbol{W}(k),\ k = 2, 3, \ldots\}$ be *strictly stationary*, where

$$\boldsymbol{W}(k) = ((U_{ij}(k), D_{ij}(k)),\ i < j,\ i, j \in V)\ ,\ k \geq 1.$$

In particular, for distinct $i < j$ in V, the sequence $\{(U_{ij}(k),\ D_{ij}(k)),\ k = 2, 3, \ldots\}$ constitutes a stationary sequence with generic marginals $(U_{ij},\ D_{ij})$. We denote by G_{ij} the cumulative distribution function (CDF) of U_{ij}.

We endow V with a time-varying graph structure by introducing a time-varying set $E(t) := \{(i,j) \in V \times V : \xi_{ij}(t) = 1\}$, $t \geq 0$, of undirected edges, where by convention we set $\xi_{ii}(t) = 0$ for each i in V and all $t \geq 0$. Providing a path from a source node to a destination node at $t \geq 0$ requires simultaneous availability of a number of communication links that are up at the time of path request and collectively provide the desired connectivity between the source and the destination. Thus, when a path request arrives, a path can be established (in principle) between nodes s and d at time $t \geq 0$, if node d is reachable from node s by a path in the graph $(V, E(t))$.

Let $\mathcal{L}_{sd}(t)$ denote the set of links in a selected path at time $t \geq 0$. For each link $\ell \in \mathcal{L}_{sd}(t)$, define $T_\ell(t)$ to be the time-to-live or excess life after time t, i.e., $T_\ell(t)$ is the amount of the time that passes from time t onward until link ℓ is torn down. The time-to-live or duration $Z_{sd}(t)$ of an established path from node s to node d using the links in $\mathcal{L}_{sd}(t)$ is defined as the amount of time that elapses from time t until one of the links in $\mathcal{L}_{sd}(t)$ goes down. This quantity is simply given by $Z_{sd}(t) := \min\left(T_\ell(t) : \ell \in \mathcal{L}_{sd}(t)\right)$.

Our results in [2], [3] state that if the number of hops $|\mathcal{L}_{sd}(t)|$ in a path is large, then under a set of mild assumptions, the distribution of $z_{sd}(t)$ can be well approximated by an exponential distribution. Moreover, the parameter of the exponential distribution λ is given by $\sum_{\ell \in \mathcal{L}_{sd}(t)} \lambda_\ell$, where λ_ℓ is the inverse of the expected duration of link ℓ, i.e., $\lambda_\ell = 1/\mathbf{m}(G_\ell)$, where $\mathbf{m}(G_\ell)$ denotes the mean of the distribution G_ℓ.

III. Proposed path selection scheme

Our earlier results [2], [3] summarized in Section II show that when the hop count along a path is large, the distribution of path duration can be well approximated by an exponential distribution and the parameter of the exponential distribution is given by the sum of the inverses of the expected durations of the links along the path. These suggest that in order to estimate the expected duration of a candidate path a source needs only the sum of the inverses of the estimated expected link durations. Moreover, due to the memoryless property of an exponential distribution, we can easily estimate the probability a path that was available at time t_0 is still available at a later time $t_1 > t_0$. This is exploited in our proposed scheme during a path recovery procedure. In the rest of the paper we assume only on-demand routing protocols.

Estimation of the exponential distribution parameter – In our proposed scheme each node maintains an average $\mathbf{LD}_{avg}$ of the link durations it experiences. Such an average can be maintained by updating the average value at the times of link teardowns using, for example, exponentially weighted moving average (EWMA):

$$\mathbf{LD}_{avg}^{new} = (1 - w)\mathbf{LD}_{avg}^{old} + w \cdot \mathbf{LD}_{new} \tag{1}$$

where $\mathbf{LD}_{avg}^{new}$ and $\mathbf{LD}_{avg}^{old}$ are the new and old average link durations, respectively, $\mathbf{LD}_{new}$ is the duration of the link that was just terminated, and $w \in (0, 1]$.

In our proposed scheme we add a field called *inverse_path_duration_* (IPD) to a path reply message. When the destination replies to a path request, it initially sets the value of IPD to zero. A neighboring node that receives the reply message first adds to the value in the IPD field the inverse of its estimate of expected duration of the link with the destination on which the reply message arrived, and then forwards it to the next upstream node. The estimate of the expected link duration is given by the node's average link duration in (1). When the next upstream node receives the message, it adds the inverse of its estimate of the expected duration of the link on which the reply message arrived, and forwards it to the next upstream node. This process repeats until the reply message reaches the source. Upon receiving the reply message, the source adds to the value of the IPD field the inverse of its estimate of the expected duration of the link on which the reply message is received, and uses the final value as the parameter of the exponential distribution. Similarly, when an intermediate node with a known path to the requested destination replies to a path request message, it specifies the inverse of its estimate of the expected duration of the path to the destination in the IPD field in the reply message and sends it to the upstream node that forwarded the request message.

Path recovery – When a link along a path in use fails, a recovery can be initiated by the node that detects the link failure or any other upstream node. Suppose a node has a cached backup path in its routing table. Then, for each cached backup path, our proposed algorithm first computes the probability the path is still available as follows: Suppose that the last time the entry was updated (hence the path was still available) was at time t_0. If the path failure takes place at time $t_1 > t_0$, then the probability the path is still available at time t_1 is approximated by

$$\exp(-\lambda \cdot (t_1 - t_0))\ , \tag{2}$$

where λ is the IPD value of the path recorded in the routing entry. Our proposed scheme selects the backup path with the smallest IPD among the ones whose probability of being available given by (2) exceeds certain threshold value γ, i.e.,

$$\exp(-\lambda \cdot (t_1 - t_0)) \geq \gamma \ . \tag{3}$$

If there is no such backup path, the node either initiates a new path discovery procedure or sends an error message to upstream node(s). The goal of this scheme is to minimize the routing overhead incurred during path discovery procedures in the event of a path failure by choosing a backup path that is likely to be available.

IV. Update of parameter γ

As described in the previous section, our proposed scheme attempts to reduce the routing overhead by switching to a backup path cached in the routing table which satisfies the condition (3) in case of a path failure. However, the probability computed by (2) is only an approximation; first, our results in [2], [3] are asymptotic results. Hence, for any finite hop count, the true distribution of path duration may not be exponential and the mean of the distribution may not equal the inverse of the IPD value. Second, the IPD values are computed using noisy measurements of the average link durations. For these reasons, the probability a backup path is still available at the time of a path failure may differ from that given by (2). In this section we discuss the problem of selecting the parameter γ so that the probability a selected backup path is available equals some desired value $\mathbf{p}_{target} > 0$.

Define $X(\gamma)$ to be the indicator function of the event that a selected backup path is available when the threshold value is set to γ. For each $\gamma \in (0, 1)$, let $G(\cdot, \gamma)$ and $m(\gamma)$ denote the *unknown* distribution function of the Bernoullie random variable $X(\gamma)$ and its mean, respectively. Clearly, it is reasonable to assume that $m(\gamma)$ is strictly increasing in $\gamma \in (0, 1)$ because the probability a selected backup path is available should increase with the threshold γ. Our problem can now be formulated as one of finding γ such that $m(\gamma) = \mathbf{p}_{target}$.

We assume that there exists a threshold $\gamma^\star \in (0, 1)$ such that $m(\gamma^\star) = \mathbf{p}_{target}$. Let $\gamma_n, n = 0, 1, \ldots,$ denote a node's estimate of $\gamma^\star$ after trying n backup paths with the initial value $\gamma_0 = \mathbf{p}_{target}$. Each time the node tries a backup path (when the estimated probability in (2) exceeds γ_n), it updates its threshold as follows:

$$\gamma_{n+1} = \gamma_n + \epsilon_n(\mathbf{p}_{target} - X_n) \ , \ n = 0, 1, \ldots \tag{4}$$

where X_n is the indicator function of the event that the selected backup path is available, and ϵ_n is a positive step size.

We can rewrite (4) as

$$\gamma_{n+1} = \gamma_n + \epsilon_n(\mathbf{p}_{target} - m(\gamma_n)) + \epsilon_n(m(\gamma_n) - X_n). \tag{5}$$

Define $\mathcal{F}_n$ to be the σ-field generated by $\{(X_i, \gamma_i), i = 0, \ldots, n-1, \gamma_n\}$. The martingale difference $m(\gamma_n) - X_n$ satisfies $\mathbf{E}\left[m(\gamma_n) - X_n | \mathcal{F}_n\right] = 0$ w.p. 1 from the definition of $m(\gamma)$.

The following results can be proved easily from well known results in stochastic approximation literature [1], [4].

Lemma 1: (i) If the step sizes satisfy $\sum_{n\geq 0} \epsilon_n = \infty$ and $\sum_{n\geq 0} (\epsilon_n)^2 < \infty$, then $\lim_{n\to\infty} \gamma_n = \gamma^\star$ w.p. 1. (ii) Suppose that the step sizes satisfy $\underline{\epsilon} \leq \epsilon_n \leq \overline{\epsilon}$ for all $n = 0, 1, \ldots,$ for some constants $0 < \underline{\epsilon} < \overline{\epsilon} < 1$. Then, for any $\delta > 0$, there exists $b := b(\delta) < \infty$ such that $\limsup_{n\to\infty} \mathbf{P}\left[|\gamma_n - \gamma^\star| \geq \delta\right] \leq b\overline{\epsilon}$.

Lemma 1(i) states that if the step sizes decrease properly, then the threshold γ_n converges to $\gamma^\star$ almost surely. Lemma 1(ii) implies that if the step size is fixed to be a sufficiently small constant, then γ_n will asymptotically stay in a small neighborhood around $\gamma^\star$ with a high probability.

V. Simulation

We implemented our proposed scheme described in Section III in the AODV routing protocol and evaluated its performance against (unmodified) AODV and min-hop routing protocols.

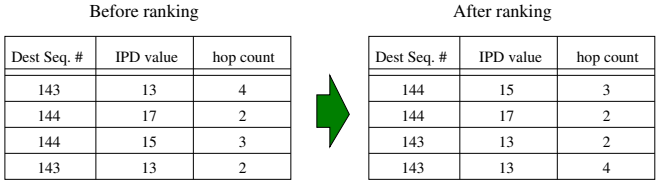
A. Implementation in AODV

Routing table – Under the new protocol a node creates and maintains a route entry for each known destination node.[1] However, unlike in the AODV protocol, the node can keep up to k paths instead of a single path, where k is a design parameter. In our simulation the value of k is set to three. The information regarding each path is recorded in a subentry consisting of five fields: (i) destination sequence number, (ii) next hop to the destination, (iii) hop count, (iv) Inverse Path Duration (IPD), and (v) time stamp. The IPD field contains the the sum of the inverses of expected link durations reported in a path reply message during path discovery. The time stamp field contains the time of last update when the path was still available. We require that the next hop to the destination of these paths be different to minimize redundant path information. If there is more than one path available, one of them is selected as the primary path, and the others are cached as backup paths in case the primary path fails.

The paths in a route entry are ranked first based on the destination sequence numbers, and then based on the IPD values. Ties are broken using the hop counts. To be precise, when more than one path to a destination are discovered, the paths are first ranked by decreasing destination sequence number. If two or more paths have the same sequence number, then the one with a smaller IPD value takes a higher preference as it has a larger *expected* duration than the others. Finally, if the first two are the same, the path with a smaller hop count is preferred. This is shown in Fig. 1. The path with the highest preference is used as the primary path for routing packets when needed. When the primary path breaks down, the path with the second highest preference is selected as an alternate path if its probability computed by (2) exceeds γ.

A path *request* message contains (i) source ID and its sequence number, (ii) broadcast ID, (iii) destination ID and

[1]The routing protocol running at a node may not be aware of node's neighbors at the beginning when the route entries are empty.



Dest Seq. #	IPD value	hop count
143	13	4
144	17	2
144	15	3
143	13	2

Dest Seq. #	IPD value	hop count
144	15	3
144	17	2
143	13	2
143	13	4

Fig. 1. Ranking of discovered paths.

its sequence number, and (iv) hop count to the source. The hop count is initially set to one. Each path request message is distinguished by its source ID and broadcast ID maintained by the source node. The broadcast ID is increased by one each time the source generates a new path request message. The source sequence number is used by intermediate nodes for updating route entry associated with the *source*.

When a node receives a path request message, it first updates the routing information to the source using information in the request message. In particular, if there is a subentry with the neighbor node that forwarded the request message as the next hop, it updates the source sequence number and the hop count. If necessary, it re-ranks the paths to the source based on the new sequence number and the hop count after the update. Then, if the node had received another copy of the same request message, it terminates the message. Otherwise, if it is not the requested destination node in the message and has no routing information to the destination, it increases the hop count in the message and rebroadcasts the message to its neighbors.

Path reply message – If an intermediate node has a route entry for the destination with the destination sequence number no smaller than the destination sequence number in the request message, it generates a path *reply* message. The reply message contains (i) destination ID, (ii) destination sequence number, (iii) IPD value, and (iv) hop count. The values of these fields are copied from the subentry for the primary path to the destination node. The reply message is then sent to the neighbor node that forwarded the request message, and the request message is terminated.

If a copy of the path request message reaches the destination node, the destination first increases its sequence number to the maximum of its current sequence number and the destination sequence number in the request message. It then generates a reply message with the same four fields in the reply message generated by an intermediate node with the initial value of the IPD and the hop count set to zero. If the destination node receives multiple copies of the same path request message from different neighbor nodes, it can generate more than one reply message.

When an intermediate node receives a reply message, it first adds to the IPD value in the reply message its estimate of the inverse of the expected duration of the link on which the reply arrived. This estimate is given by its average link duration. Then, it updates its routing information to the destination; it creates a temporary subentry for the newly discovered path to the destination with the information provided in the reply message. Then, if the new path is better than the k-th ranked path in the entry for the destination, it re-ranks the paths, including the newly discovered path, and inserts the new subentry in the entry for the destination. Otherwise, it discards the temporary subentry. If the new path is selected as the primary path to the destination, the node broadcasts the reply message, indicating a better path to the destination is discovered.[2]

Finally, when the source receives a reply message it first updates the hop count and the IPD value using its estimate of the expected duration of the link on which the reply arrived, and then its route entry for the destination. Data transmission can begin as soon as the first path to the destination is discovered. However, if another path discovered later is deemed better than the current primary path, the source may switch its primary path to the destination.

Path failure – When a link failure occurs, the node that detects the failure attempts a local recovery as follows: For each backup path it calculates the probability that the backup path is still available using the estimated IPD value of the path according to (2). Suppose that there exists at least one backup path that satisfies the condition (3). Then, the backup path with the smallest IPD value satisfying (3) is selected as an alternative path, and data packets are routed through it immediately without further checking the availability of the path. If the backup path turns out to be unavailable, a path recovery procedure is initiated and a *route error* message is broadcast. The error message contains a list of destination nodes no longer reachable after the link failure and the *local_recovery_* (LR) field set to 1. If no backup path satisfies the condition (3), a *route error* message is generated with the LR field set to 0.

Each node that receives an error message checks if the LR field has been to set to 0. If so and a primary path is affected by the link failure, the node can perform a local recovery described above.

B. Simulation results

In this subsection we evaluate the performance gain (in path duration) from our scheme outlined in the previous subsection. For comparison purposes we run the simulation with (i) AODV, (ii) min-hop routing, and (iii) our proposed scheme using the same mobility file and the same path request patterns.

The simulation is run with 200 nodes moving in a 2 km $\times$ 2 km rectangular region. The Random Waypoint mobility model is employed. The transmission range of the nodes is set to 250 m. In order to create a scenario with heterogeneous nodes that experience diverse link durations in the field (e.g., soldiers vs. tanks or jeeps), we introduce two classes of nodes. The speed of a class 1 and class 2 node is uniformly distributed in [1, 5] m/s and [10, 30] m/s, respectively. Slower moving class 1 nodes in general experience longer link durations than faster

[2]The quality of a discovered path can depend on many factors, including available bandwidth, congestion level, etc. in addition to the expected duration. However, for the purpose of discussion in this section we only consider the expected duration and hop count.

moving class 2 nodes in our simulation due to the same fixed transmission range.

Each run of simulation is for 1,200 seconds, and a total of 126 runs are carried out with different random seeds.[3] However, in order to reduce the effects of transient period data are collected only in the last 800 seconds of each run. A total of approximately 5,000 connections are set up between randomly selected source and destination nodes. The interarrival times of connection requests are given by independent and identically distributed rvs, each of which is a sum of 5 seconds and an exponential rv with a mean 15 seconds. Each connection request generates a path request message, triggering a path discovery phase. Therefore, in the last 800 seconds of each run we generate on the average 40 path request messages.

We simulate three different scenarios by varying the number of class 1 nodes (hence class 2 nodes as well) to illustrate the benefits of our scheme and a trend that emerges. We first begin with 140 class 1 nodes and increase it to 160 and then to 180. Our scheme is run under two different modes: In the first mode, each node maintains a single average link duration for all neighbor nodes. In the second mode each node classifies the neighbors and maintains two separate averages - one for class 1 neighbors and the other for class 2 neighbors. Classifying neighbors and maintaining two separate averages allows nodes to obtain more accurate estimates of expected link durations based on the type of the neighbors. Hence, it improves the accuracy of the estimated IPD values during path discovery.

The CDFs of link duration under our scheme for the 160 vs. 40 scenario are shown in Fig. 2. In the first mode, we plot the distributions seen by class 1 nodes and class 2 nodes. In the second mode, we plot the distributions of the links between (i) two class 1 nodes, (ii) between a class 1 node and a class 2 node, and (iii) two class 2 nodes. As expected, in the first mode link durations seen by class 1 nodes are much larger than those seen by class 2 nodes in the usual stochastic order. However, note that the discrepancy in the link duration distribution (a) between two class 2 nodes and (b) between a class 1 node and a class 2 node is not very large.

The CDFs of the path duration under AODV and our scheme (both with and without separate averages) are plotted in Fig. 3. It is clear from Fig. 3 that the CDF of path duration under our scheme lies below that under AODV. In other words, path durations under AODV are stochastically smaller than path durations under our scheme. This indicates that our scheme does a better job of selecting paths with longer durations than AODV even without separate averages of link durations. The median values of path durations are given in Table I. In the first two scenarios there is a 60 percent increase in the median values over AODV when nodes maintain separate averages.

AODV always selects the first discovered path. Thus, it does not attempt to avoid paths that traverse many class 2 nodes. When the number of class 2 nodes is 60, due to a large number of class 2 nodes it is beneficial to minimize the number of links involving class 2 nodes along a selected path. This is because the average duration of the links between two class 1 nodes

[3]This is done to reduce the size of the mobility file needed in ns-2 simulation.

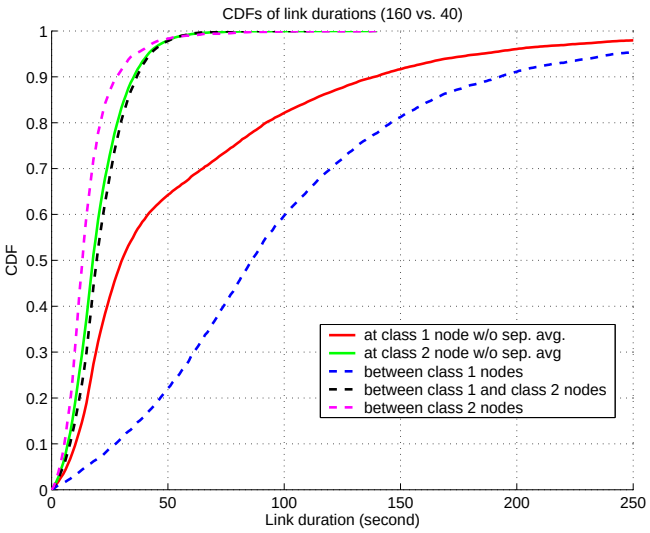


Fig. 2. CDFs of link durations.

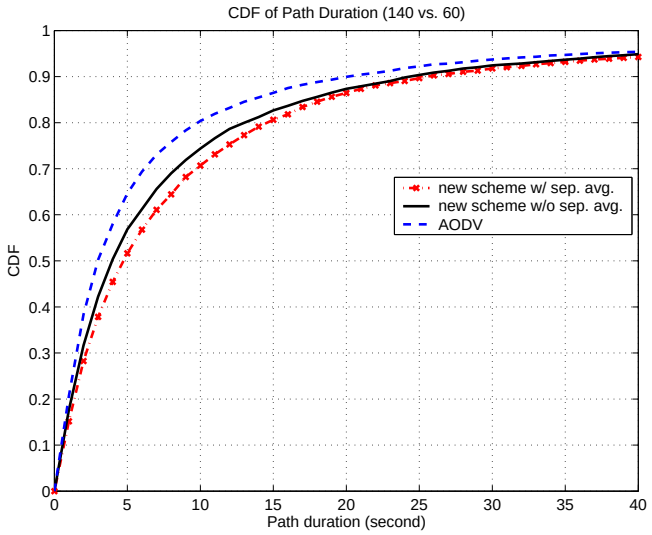
TABLE I
MEDIAN VALUES OF PATH DURATIONS

class 1 & 2 nodes	AODV	w/o sep. avg.	w/ sep. avg.
140 & 60	2.98	3.97	4.76
160 & 40	3.68	5.80	5.88
180 & 20	5.90	8.00	8.10

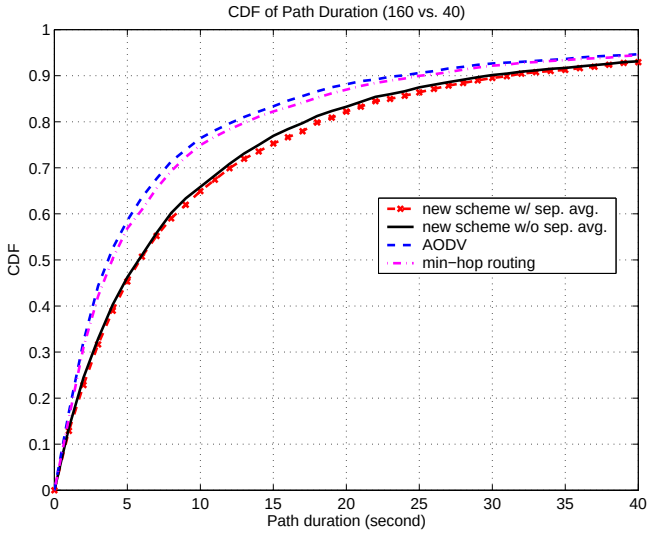
is much larger than that of other links involving a class 2 node (see Fig. 2 for an example[4]), and explains the difference between AODV and our scheme without separate averages. For a similar reason, although the benefit is not quite as large, maintaining separate averages at the nodes helps avoid the link(s) between two class 2 nodes along a selected path (which is not possible without maintaining separate averages) and provides further benefit. However, as the number of class 2 nodes is reduced to 40 and then to 20, the benefit of maintaining separate averages diminishes. This is because by avoiding links involving class 2 nodes, our scheme can avoid most of links between two class 2 nodes as they become more scarce with decreasing number of class 2 nodes. Furthermore, the benefit from our algorithm decreases when the number of class 2 nodes is reduced from 40 to 20. This follows from the observation that when there are only 20 class 2 nodes, most of the links do not have a class 2 node as a terminal node and hence many of the paths selected by AODV do not traverse class 2 nodes even without attempting to do so.

Frequent path failures can degrade applications' performance considerably or even render them ineffective. Therefore, minimizing the number of short-lived paths can significantly improve perceived performance. From this viewpoint it is more interesting to look at the probability $\mathbf{P}\left[\text{path duration } \leq x\right]$ for small values of x. This is summarized in Table II for $x = 3$ seconds. When compared to the numbers with separate averages, the numbers for AODV are 27.4 to 40.1 percent larger, which indicates significant

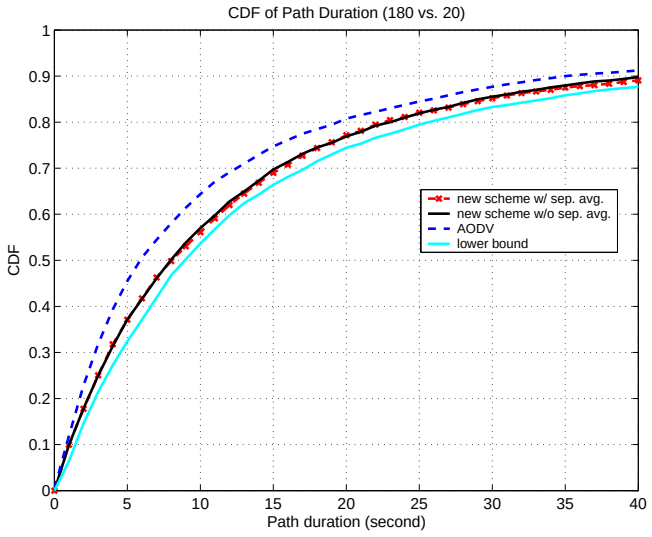
[4]Although Fig. 2 shows the CDFs of link duration for the 160 vs. 40 scenario, the qualitative nature of the plot is similar for 140 vs. 60 scenario.



(a)



(b)



(c)

Fig. 3. CDFs of path duration.

improvement by our scheme.

TABLE II

$\mathbf{P}$[PATH DURATION $\leq$ 3 SECONDS]

class 1 & 2 nodes	AODV	w/o sep. avg.	w/ sep. avg.
140 & 60	0.5024	0.4224	0.3784
160 & 40	0.4436	0.3288	0.3166
180 & 20	0.3179	0.2496	0.2496

We also run the simulation with min-hop routing for the 160 vs. 40 scenario. This is done by modifying our scheme and selecting a path based on the hop count (as opposed to the IPD value in our scheme). The CDF of path duration under the min-hop routing is shown in Fig. 3(b). Although not shown here, the hop counts of the paths selected by the min-hop routing are stochastically smaller than those of our scheme. However, as shown in Fig. 3(b) the path durations are stochastically larger under our scheme. In fact, the difference between the min-hop routing and AODV is very small.

In order to evaluate the performance of our scheme we also compare its performance for the 180 vs. 20 scenario to that of the min-hop routing with only class 1 nodes. Since nodes are homogeneous in the latter, the min-hop routing is equivalent to our algorithm (because the average link durations seen by the nodes should be approximately the same at steady state) and selects the paths with the largest expected durations. The CDF of path duration under this scenario is shown in Fig. 3(c) as 'lower bound'. Note that this lower bound may not be achievable since no class 2 nodes are used in the case. Nevertheless, the CDF under our scheme is very close to that of the lower bound. This demonstrates that our algorithm does a good job of selecting the paths with the largest expected durations by avoiding links with potentially short excess lives.

One thing to notice from Fig. 3 and Table I is that the link durations decrease rapidly with the number of class 2 nodes (hence the number of links involving a class 2 node). This suggests that the path durations are often determined by the excess life of a link with a class 2 node. This is another evidence that corroborates our claim that avoiding nodes with shorter average link durations is beneficial in selecting longer lasting paths in MANETs.

REFERENCES

[1] V. S. Borkar and S. P. Meyn, "The O.D.E. method for convergence of stochastic approximation and reinforcement learning," *SIAM J. Control Optim.*, Vol. 38(2), pp. 447-469, 2000.

[2] Y. Han, R. J. La, A. M. Makowski, and S. Lee, "Distribution of path durations in mobile ad-hoc networks - Palm's theorem to the rescue," *to appear in Computer Networks (Special Issue on Network Modeling and Simulation).*

[3] Y. Han, R. J. La, and H. Zhang, "Path selection in mobile ad-hoc networks and distribution of path duration," *Proc. of IEEE Infocom 2006*, April 2006.

[4] H. J. Kushner and G. George Yin, *Stochastic Approximation and Recursive Algorithms and Applications*, 2nd ed., Springer, New York, New York, 2003.

[5] D. B. Johnson, and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing*, pp. 153-181, 1996.

[6] C. E. Perkins, *Ad-hoc Networking*, Addison-Wesley Longman, Incorporated, 2000.

[7] C. Perkins and E. M. Royer, "Ad hoc on-demand distance vector routing," in Proceedings of the second annual IEEE workshop on mobile computing systems and applications, pp. 90-100, February 1999.