A Model of Topology Cache in Reactive Routing Protocols for MANETs*

Andres Medina
Dept. of Electrical and Computer Eng.
University of Delaware
medina@ece.udel.edu

Stephan Bohacek
Dept. of Electrical and Computer Eng.
University of Delaware
bohacek@ece.udel.edu

ABSTRACT

In a reactive routing protocol such as DSR, when a route search message is flooded, a large number of nodes learn the path to the originator of the search. Moreover, each of these nodes learns paths to all upstream nodes along the route to the originator. Thus, during a route search, a large amount of topology information is distributed and stored in cache. This information is then used to reduce the number of hops future route search messages must travel. In this paper a model of the impact of topology cache is developed. The model allows the impact of cache to be estimated for a wide range of node speeds, node densities, network sizes, and values of cache timeout. When compared to simulation, the model predicts the number of hops that the route search message spreads within 10%. This model allows a wide range of analysis to be performed. For example, the paper includes an investigation of optimal cache timeout and an investigation of the impact of cache in highly mobile environments.

Categories and Subject Descriptors

C.2.2 [Computer-Communication Networks]: Network Protocols—Routing protocols

1. INTRODUCTION

Analytic models have some advantages over simulations for performance evaluation. Specifically, analytic models can be used to efficiently explore a large space of system and environmental parameter, whereas a similar exploration with simulation-based performance evaluation would required considerable computationally resources. On the other hand, models of complicated protocols can be difficult to develop. As a result, much of the performance analysis of routing protocols for mobile ad hoc networks has relied on simulation. However, this paper develops a model of a DSR-style topology cache [1], a complicated and yet critical part of some reactive routing protocols. The performance of topology cache depends on a wide range of parameters, including the number of nodes in the network, the density of the nodes (in relation to the transmission range), the size and shape of the region covered by the network, the rate at which route requests are made, the cache timeout, and the node speed. Consequently, previous simulation-based performance evaluation of topology cache has not explored the entire parameter space.

The model developed here can be used in several ways. For example, the optimal value of cache timeout and the impact of cache can be determined. Topology cache is an important distinction between source-based routing protocols such as DSR and table-based protocols such as AODV. Hence, model of the impact of cache help determine when and why DSR out performs AODV.

While the main goal of this paper is to develop a model of cache, several useful sub-models were also developed. For example, Section 8 presents a new model for the link and path lifetime. In contrast to prior work [2], modeling link lifetime as independent and exponentially distributed was found to yield a low quality fit. Instead, a novel technique is developed that scales the link lifetime distribution with the speed of nodes. This paper also presents models that relates distance and hops in low density networks.

The paper proceeds as follows. In the next section, an overview of the scenario studied is detailed. Section 3-8 develop the model of topology cache. Section 3 provides an overview of the model, while Sections 4-8 develop the require sub models. Section 9 compares the model to simulation and shows that the number of hops that a route search must propagate to reach a node with the desired path is modeled with an accuracy of 10%. Section 10 provide a few of the insights that can be gleaned from the model. And finally, Section 11 provides some concluding remarks.

Before proceeding, a few notes on notation are useful. The probability density function (pdf) of a random variable X will be denoted as p_X, whereas its cumulative distribution function (cdf) will be written P_X. When the pdf of a random variable is known, the cdf is also known and vise versa. The conditional pdf of X given Y is denoted $p_{X|Y}$. Parameters of a pdf are specified after a semicolon, e.g., $p_{X|Y}(x|y;m)$, m is a system parameter.

2. SCENARIO UNDER CONSIDERATION

Under DSR, when a route is required, a route request ($RREQ$) message is flooded until either the desired destination is reached or the $RREQ$ message reaches a node with a path to the desired destination in cache. While there are a wide range of methods for flooding $RREQ$s (e.g., see [3]), here we assume that expanding ring flooding is used. However, it is straightforward to extend the model to other flooding techniques. In any case, when a $RREQ$ reaches

*This work was prepared through collaborative participation in the Collaborative Technology Alliance for Communications and Networks sponsored by the U.S. Army Research Laboratory under Cooperative Agreement DAAD19-01-2-0011. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

WICON'08, November 17-19, 2008, Maui, Hawaii, USA.

a node, the node determines whether the *RREQ* has been processed before. If that is the case, the *RREQ* is discarded, otherwise, the node checks if it has a route to the desired destination in its topology cache. If so, then the node send a route reply (*RREP*) message to the originator of the *RREQ* and the search is complete. If the node does not have the desired path in cache, then it will save the path back to the originator into the topology cache. This path also includes paths to each node along the path. After saving this topology information, the node appends its address to the path back to the originator in the *RREQ*, and after a random delay to avoid collisions, broadcast the modified *RREQ*. It is important to emphasize that each *RREQ* received and processed by a node results in the node saving some topology information. Consequently, flooding *RREQ* messages results in a large amount of topology information being disseminated.

In this paper, it is assumed that nodes move according to the random way-point mobility model [4], and thus, nodes are distributed according to the steady state distribution of the random way-point model [5]. It is assumed that all radios have the same transmission range, denoted by R. Furthermore, it is assumed that all transmissions are received error free by nodes that are R from the transmitter. However, all distances are normalized by the transmission range, i.e., $R = 1$. Similarly, time is normalized by the cache timeout, and hence, node speed is normalized by transmission ranges per cache timeout.

3. PROBABILITY DENSITY FUNCTION OF THE NUMBER OF HOPS A RREQ PROPAGATES

When a source node $\mathcal{S}$ originates a route search message (i.e., a *RREQ* message) to find destination node $\mathcal{D}$, the number of hops the *RREQ* propagates, depends on whether a path to $\mathcal{D}$ is stored in the cache of the nodes that the *RREQ* packet reaches. If the search is done in a system with no cache, the *RREQ* propagates all the way to $\mathcal{D}$. However, if the *RREQ* reaches a node with a path to $\mathcal{D}$ stored in cache, then the route search can be confined to a smaller set of nodes, reducing overhead. Let K_S be the number of hops that the route search propagates until it reaches $\mathcal{D}$ or a node that has a path to $\mathcal{D}$ stored in cache. K_S is a random variable that depends on the distance D_S from the source to the destination and the current cache information in the system. The central goal of this paper is to estimate the distribution of K_S.

Consider a sequence of route searches defined by $\left\{\mathcal{S}^i,\ \mathcal{D}^i,\ t^i\right\}$, where $i = 1$ denotes the first route search after an empty cache system. $\mathcal{S}^i$and $\mathcal{D}^i$ are the source and destination of the i-th route search, respectively, and t^i is the time at which the i-th route search is made. Assume all nodes in the network use the same cache timeout, TO. Any cache information older than TO seconds is considered invalid and is not used. To take advantage of a previous route search, the route request i, has to be made no later than TO seconds after the $(i-1)$-th route search, i.e., $t^i - t^{i-1} \leq TO$, for all i. Define as an "old source" ($\mathcal{OS}$) of the i-th route search, any previous source that made its route request in $\left(t^i - TO, t^i\right)$.

Figure 1 illustrates a scenario where a route search results in a path being found in cache. An old source (marked OS in Figure 1) originated a previous route search. Node $\mathcal{D}^i$ forward the *RREQ* on behalf of OS. Consequently, the nodes in the shaded region learn paths to $\mathcal{D}$, and store these paths in their cache. Later, node $\mathcal{S}^i$ originates a search for a path to node $\mathcal{D}^i$. The *RREQ* of this search propagate K_S hops before reaching a node that has a path to $\mathcal{D}^i$. Note that the distance (in meters) that the *RREQ*s propagate is denoted by $D(K_s)$, as shown in the Figure. Also, the distance between $\mathcal{S}^i$ and $\mathcal{D}^i$ is denoted by D_S. Clearly, if $D(K_S) = D_S$, then the *RREQ* reaches $\mathcal{D}^i$ and cache is not used.

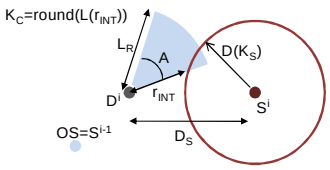


Figure 1: Scheme of i-th search. Nodes in shaded area have information about D^i spread by the route search of $OS = S^{i-1}$. Circle denotes search area of S^i.

We begin by considering the case where $t^i - t^{i-1} < TO$ and $t^i - t^{i-2} > TO$, for all i, i.e., the i-th route search is performed with the information of exactly one previous old search in the cache. The distribution of K_S^i, depends on D_S^i and on K_S^{i-1}, the number of hops that the *RREQ* message of the previous route search originated by $\mathcal{S}^{i-1}$ propagated. Define $G(k_S, k_{OS}, d_S) := P\left(K_S \leq k_S \middle| K_{OS} = k_{OS}, D_S = d_S\right)$, where K_{OS} is the number of hops that the search of the old source propagated. Letting $P^i_{K_S|D_S}$ be the conditional distribution of K_S^i, we have

$$P^i_{K_S|D_S}\left(k_S \middle| d_S\right) = \sum_{k_{OS}=0} G\left(k_S, k_{OS}, d_S\right) p^{i-1}_{K_S}\left(k_{OS}\right), \quad (1)$$

where

$$P^i_{K_S}\left(k_S\right) = \int_{d_S} P^i_{K_S|D_S}\left(k_S \middle| d_S\right) p_{D_S}\left(d_S\right) dd_S. \quad (2)$$

and p_{D_S} is the pdf of the distance between the source and destination. In steady state, the distribution of K_S satisfies

$$P_{K_S}\left(k_S\right) = \int_{D_S} \left(\sum_{k_{OS}=0} G\left(k_S, k_{OS}, d_S\right) p_{K_S}\left(k_{OS}\right) \right) dd_S.$$

Assuming it exists, $P_{K_S} = \lim_{i\to\infty} P^i_{K_S}$ where $p^0_{K_S}(k) = \delta_{\{k=0\}}$, i.e., there were no previous searches to fill cache[1]. Thus, if G and p_{D_S} are known, then P_{K_S} can be computed by iterating (1) and (2).

In the more general case, we assume that topology information from exactly N_{OS} old searches is the cache before every route search. In this case, the route search propagates until it reaches the destination or it reaches a node that has a path to $\mathcal{D}$, where this path was saved into cache during one of the previous N_{OS} route searches. In effect, each previous route search fills a separate cache. Let each node place any topology information learned by the m-th previous route search in cache $\mathcal{C}_m$. For a fixed m, the probability that no nodes within k_S hops of $\mathcal{S}$ have a path to $\mathcal{D}$ in $\mathcal{C}_m$ is $\left(1 - \sum_{k_{OS}=0} G\left(k_S, k_{OS}, D_S\right) p_{K_{OS}}\left(k_S; N\right)\right)$, where $P_{K_{OS}}$ is the distribution of K_{OS} for each of the previous searches. Assuming the previous route searches are independent, the probability that no node within k_S hops

[1]The existence and uniqueness of this limit is not investigated, but computational experiments have indicated that it always exists.

of $\mathcal{S}$ has a path to $\mathcal{D}$ in any of the N_{OS} caches is $\left(1-\sum_{k_{OS}=0} G\left(k_S,k_{OS},D_S\right) p_{K_{OS}}\left(k_S;N\right)\right)^{N_{OS}}$.

Therefore, as in the single old source case, in steady state $p_{K_S|D_S}\left(K_S|\, D_S;N_{OS}\right)$, the conditional pdf of K_S given D_S, satisfies

$$P_{K_S|D_S}\left(k_S|\, d_S;N_{OS}\right)=1-\left(1-\sum_{k_{OS}=0} G\left(k_S,k_{OS},d_S\right) p_{K_S}\left(k_S;N_{OS}\right)\right)^{N_{OS}} \quad (3)$$

where

$$p_{K_S}\left(k_S;N_{OS}\right)=\int_{d_S} p_{K_S|D_S}\left(k_S|\, d_S\right) p_{D_S}\left(d_S\right) dd_S, \quad (4)$$

Furthermore, if G and p_{D_S} are known, then we can compute such a P_{K_S} by letting $P_{K_S}=\lim_{i\to\infty} P^i_{K_S}$ where

$$P^i_{K_S|D_S}\left(k_S|\, d_S;N_{OS}\right)=1-\left(1-\sum_{k_{OS}=0} G\left(k_S,k_{OS},d_S\right) p^{i-1}_{K_S}\left(k_S;N_{OS}\right)\right)^{N_{OS}} \quad (5)$$

with

$$p^i_{K_S}\left(k_S\right)=\int_{d_S} p^i{}_{K_S|D_S}\left(k_S|\, d_S\right) p_{D_S}\left(d_S\right) dd_S, \quad (6)$$

and $p^0_{K_S}\left(k_s;N_{OS}\right)=\delta_{\{k_s=0\}}$.

Now consider the case where all nodes are moving at speed s. Due to node mobility, links may break, and hence the topology information stored in node caches may become stale. Consequently, a route search might return a broken path. When a source of a route search detects that the just found path is broken, various strategies can be followed to obtain a valid path. In this paper, we assume that if $\mathcal{S}$ initially find an invalid path after the *RREQ* has propagated K_S hops, it will restart its search for $\mathcal{D}$ at K_S+1 hops and will require that the reply message comes directly from $\mathcal{D}$. That is, if the first search fails, then cache is not used for the second search and the *RREQ* travels all the way to $\mathcal{D}$. Under this strategy, K_S is the distance that the *RREQ* travels in the first search or the distance that *RREQ* must travel in order to reach $\mathcal{D}$. We denote by $K1_S$ the distance that the *RREQ* travels during the first search and by $K2_S$ the distance the *RREQ* must travel to reach $\mathcal{D}$. Letting CP be the event that the first route search discovered a correct path, then $K_S=K1_S$ if CP occurs, and $K_S=K2_S$ if CP does not occur. Therefore,

$$\begin{aligned} &p_{K_S|D_S}\left(k_S|\, d_S;s,N_{OS}\right)=p_{K_S,CP|D_S}\left(k_S,CP|\, d_S;s,N_{OS}\right)\\ &+p_{K_S,CP|D_S}\left(k_S,\overline{CP}\middle|\, d_S;s,N_{OS}\right) \quad (7)\\ &=p_{K1_S|D_S}\left(k_S|\, CP,d_S;s,N_{OS}\right)P\left(CP|\, k_S,d_S;s,N_{OS}\right)\\ &+p_{K2_S|CP,D_S}\left(k_S|\,\overline{CP},d_S;s,N_{OS}\right)P\left(\overline{CP}\middle|\, d_S;s,N_{OS}\right)\\ &=p_{K1_S|D_S}\left(k_S|\, d_S;s,N_{OS}\right)P\left(CP|\, k_S,d_S;s,N_{OS}\right)\\ &+p_{K_S|D_S}\left(k_S|\, d_S;0,0\right)P\left(\overline{CP}\middle|\, d_S;s,N_{OS}\right)\end{aligned}$$

The third equality relies on the independence of $K1_S$ with respect to CP and on $p_{K2_S|CP,D_S}\left(k_S|\,\overline{CP},d_S;\ \ s,N_{OS}\right)=p_{K_S|D_S}\left(k_S|\, d_S;0,0\right)$. This equality is true since the distribution of the number of hops that the *RREQ* travels when the first search fails is the same as the distribution the *RREQ* travels when there are no old sources, in which case, the speed is not relevant, and hence, can be set to zero. The probability $p\left(CP|\, k_S,d_S;s,N_{OS}\right)$ will be discussed in Section 8, and

$$p\left(\overline{CP}\middle|\, d_S;s,N_{OS}\right)=1-\sum_{k_S} p_{K_S|D_S}\left(k_S|\, d_S;s,N_{OS}\right)P\left(CP|\, k_S,d_S;s,N_{OS}\right).$$

Following the same approach as use to derive (3), we have

$$P_{K1_S|D_S}\left(k_S|\, d_S;s,N_{OS}\right)=1-\left(1-\sum_{k_{OS}=0} G\left(k_S,k_{OS},d_S\right) p_{K_S}\left(k_S;s,N_{OS}\right)\right)^{N_{OS}} \quad (8)$$

Therefore, we require distributions $P_{K1_S|D_S}\left(k_S|\ \ d_S;s,N_{OS}\right)$ and $P_{K_S|D_S}\left(k_S|\ \ d_S;s,N_{OS}\right)$ that satisfy (7) and (8). Again, an iterative approach is taken. If G, p_{D_S}, and $P\left(CP|\ \ k_S,d_S;s,N_{OS}\right)$ are known, we compute $P_{K_S|D_S}\left(k_S|\ \ d_S;s,N_{OS}\right)=\lim_{i\to\infty} P^{i+1}_{K_S|D_S}\left(k_S|\, d_S;s,N_{OS}\right)$ by iterating

$$P^{i+1}_{K1_S|D_S}\left(k_S|\, d_S;s,N_{OS}\right)=1-\left(1-\sum_{k_{OS}=0} G\left(k_S,k_{OS},d_S\right) p^i_{K_S}\left(k_S;s,N_{OS}\right)\right)^{N_{OS}} \quad (9)$$

and

$$\begin{aligned} p^{i+1}_{K_S|D_S}\left(k_S|\, d_S;s,N_{OS}\right)&=p^{i+1}_{K1_S|D_S}\left(k_S|\, d_S;s,N_{OS}\right)\times\\ &p\left(CP|\, k_S,d_S;s,N_{OS}\right)+p_{K_S|D_S}\left(k_S|\, d_S;0,0\right)\\ \times\left(1-\sum_{k_S} p^{i+1}_{K_S|D_S}\right.&\left.\left(k_S|\, d_S;s,N_{OS}\right)p\left(CP|\, k_S,d_S;s,N_{OS}\right)\right),\end{aligned}$$

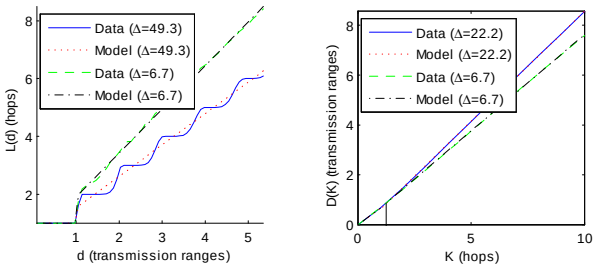
with $p^0_{K_S}\left(k_S;s,N_{OS}\right)=\delta_{\{k_s=0\}}$.

In summary, the distribution of the number of hops the *RREQ* travels can be computed once G, p_{D_S} and $P\left(CP|\ \ k_S,d_S;s,N_{OS}\right)$ are known. To this end, the following sections are organized as follows. Section 4 describes a model that relates the distance between two nodes to the length of the shortest path that communicates them. Section 5 presents a model that describes the region covered in a route search in terms of K_S. In section 6 a model that describes the spread of topology information is presented. This model is then used in section 7 to compute matrix G. Section 8 describes a model for link and path lifetime.

4. NUMBER OF HOPS BETWEEN NODES SEPARATED A DISTANCE

Let $L\left(d_S\right)$ be the average number of hops in the shortest path that connects two nodes separated a distance d_S, where the distance is normalized by the transmission range. Note that the minimum value of $L\left(d_S\right)$ is the smallest integer bigger than d_S. Such a path is composed by nodes separated by nearly one transmission range. The probability of having a node in these special positions becomes larger as the node density increases. Figure 2(a) shows the average value of $L\left(d_S\right)$ found from simulation, for a network with nodes having an average degree $\Delta=49.3$ (solid line) and $\Delta=6.7$ (dashed line). An approximately linear relationship between $L\left(d_S\right)$ and d_S is observed. Specifically $L\left(d_S\right)$ can be approximated using

$$L\left(d_S\right)\approx\begin{cases}1, \text{if } d_S\le 1\\ m_l\left(\Delta\right)d_S+b_l\left(\Delta\right), \text{if } d_S>1\end{cases}. \quad (10)$$



(a) (b)

Figure 2: (a) $L(d)$ vs d. Data and approximation values for $\Delta = 49.3$ and $\Delta = 6.7$. (b) $D(k)$ vs k. Data and approximation values for $\Delta = 22.2$ and $\Delta = 6.7$.

The accuracy of this approximation is illustrated in Fig. 2(a). The parameters used are $(m_l, b_l) = (1.0858, 0.4519)$ for $\Delta = 49.3$ and $(m_l, b_l) = (1.5115, 0.4074)$ for $\Delta = 6.7$.

Figure 3 show several values of m_l and b_l. These values were calculated by minimizing the square of the error between the model (10) and simulations. Note that as $\Delta \longrightarrow \infty$, a single transmission will reach exactly one transmission range, and a distance of between $k-1$ and k transmission ranges can be reached via k hops. Thus, as $\Delta \to \infty$, $L(d)$ becomes a staircase function, and hence $\lim_{\Delta\to\infty} m_l(\Delta) = 1$ and $\lim_{\Delta\to\infty} b_l(\Delta) = 0.5$. Models that meet these constraints are

$$m_l(\Delta) = \frac{c_{ml}}{\Delta} + 1$$

and

$$b_l(\Delta) = \begin{cases} \frac{b_{ml}}{\Delta} + 0.5 \text{ for } \Delta > 11 \\ 0.31 \text{ otherwise.} \end{cases}$$

The values of c_{ml} and b_{ml} that minimize the square error between the calculated values of m_l and b_l and the modele values are 3.5954 and -2.2393 respectively. Figure 3(a)-(b) show the quality of fit of these models.

5. AREA COVERED IN A ROUTE SEARCH

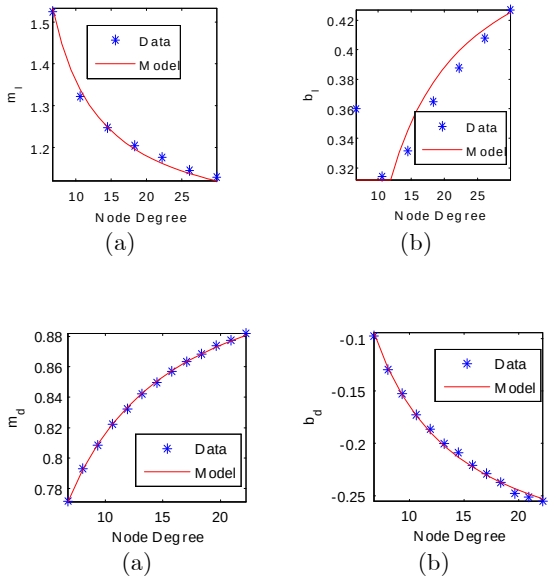
Let $D(k_S)$ be the average distance between a node originating a *RREQ* and some other node receiving this *RREQ* for the first time after the *RREQ* has traveled k_S hops. As in the case of $L(d_S)$, $D(k_S)$ is approximated using a linear model of the form

$$D(k_S) \approx \begin{cases} k_S, \text{if } k_S \leq 1 \\ m_d(\Delta)\, k_S + b_d(\Delta), \text{if } k_S > 1 \end{cases}, \qquad (11)$$

where $D(k_S)$ is normalized by the transmission range, i.e., $D(k_S) \times$ Transmission Range is distance. Figure 2(b) shows the average value of $D(k_S)$ found from simulations along with the model (11) for $\Delta = 22.2$ and $\Delta = 6.7$. Here the parameters $m_d(\Delta)$ and $b_d(\Delta)$ were found by minimizing the square error between $D(k_S)$ found from simulation and the model.

Figure 3(c)-(d) show the calculated value of $m_d(\Delta)$ and $b_d(\Delta)$ for a wide range of node densities. These calculated values are well approximated by

$$m_d(\Delta) = \frac{\Delta + 1}{1.0548(\Delta + 1) + 1.8712} \qquad (12)$$



(a) (b)

Figure 3: (a) $m_l(\Delta)$ vs. Δ. (b) $b_l(\Delta)$ vs. Δ. (c) $m_d(\Delta)$ vs. Δ. (d) $b_d(\Delta)$ vs. Δ. stars: Computed values of parameter. solid line: Parameter approximation.

and of $b_d(\Delta)$ is:

$$b_d(\Delta) = \frac{1.8461}{\Delta + 1} - 0.3325.$$

Note that for large node densities, the incremental progress by a *RREQ* traveling k to $k+1$ hops is nearly R. Thus, we expect $\lim_{\Delta\to\infty} m_d(\Delta) = 1$. This condition is approximately met by (12). On the other hand, a *RREQ* that travels k hops will reach nodes that are a distance between $R \times (k-1)$ and $R \times k$ from the source. Thus, $b_d(\Delta) < 0$ for all Δ.

6. A MODEL OF TOPOLOGY CACHE

When route search originated by $\mathcal{OS}$ propagates K_{OS}, and K_{OS} is large, a large amount of topology information is disseminated. Consequently, it is possible that due to this route search, nodes may have paths to node $\mathcal{D}$ in cache. We model the set of nodes that have paths to $\mathcal{D}$ in cache as a consequence of the a route search originated by OS as a "pie slice" shaped region as shown in Figure 4. Specifically, Figure 4 shows the set of nodes that store a path to $\mathcal{D}$ in cache as well as a pie slice shaped region that approximates this set of nodes. This pie-slice shaped region is parameterized by the orientation, the radial length, L_R, and the aperture, A, where $L_R = 0$ implies that no nodes have a path to $\mathcal{D}$. We assume that the orientation is uniformly distributed[2], and hence must model L_R and A.

The first step in developing this model is to calculate L_R and A for various scenarios. To this end, 31640 simulations were performed. For each simulation and each k_{OS}, A and L_R were calculated so as to minimize

[2]This assumption suffers from some error when $\mathcal{D}$ is near the edge of the network, in which case the nodes that have paths to $\mathcal{D}$ tend to be closer to the edge of the network than $\mathcal{D}$. Consequently, the modeled pie slice may lie outside the network. This assumption has little impact when the network is large.

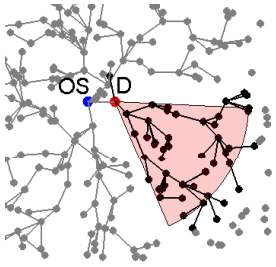


Figure 4: Nodes in black have paths to node $\mathcal{D}$ in cache as conseqence to the route search originated by node $\mathcal{OS}$. The pie slice shaped region is a model of this region.

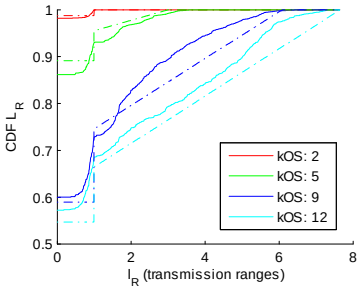


Figure 5: $P_{L_R|K_{OS}}(l_R|\,k_{OS})$ vs. l_R. Solid line: Computed values. Dashed lines: Model.

$$Err_{PIE} = \frac{\alpha \cdot bad_{in} + bad_{out}}{N_{KD}},$$

where bad_{in} is the number of nodes that are inside the pie slice and don't have a path to $\mathcal{D}$ in cache, bad_{out} is the number of nodes that are outside the pie slice and have a path to $\mathcal{D}$ in cache, and N_{KD} is the total number of nodes have a path to $\mathcal{D}$. The weighting factor α is set to 0.1 and serves to penalize large bad_{out} more than bad_{in}. Figures 5 and 6 show the cumulative probability distribution of L_R and A found from 4520 simulations for an average node degree $\Delta = 6.7415$ and various values of K_{OS}.

6.1 Modeling the radial length L_R

We model the distribution of L_R as follows. Let the size of the network be $2l_X \times 2l_Y$. We assume $0 \leq L_R \leq l_{R\max}$, where $l_{R\max}(k_{OS}) = \min\left\{\sqrt{l_X^2 + l_Y^2}, D(k_{OS}-1)\right\}$. Note that if $\mathcal{D}$ is in the middle of the network, then $L_R \leq \sqrt{l_X^2 + l_Y^2}$. However, if $\mathcal{D}$ is near to the edge, it is possible that $L_R > \sqrt{l_X^2 + l_Y^2}$, in fact, it $\mathcal{D}$ is in a corner of the network, then it possible that $L_R = 2\sqrt{l_X^2 + l_Y^2}$. On the other hand, if $\mathcal{D}$ is not in the center, then the average distance from $\mathcal{OS}$ and $\mathcal{D}$ increases, and hence for a fixed K_{OS}, we find that L_R tends to decreases as $\mathcal{D}$ is moved further from the center. Thus, $L_R \leq \sqrt{l_X^2 + l_Y^2}$ provides a reasonable approximation, with error only occurring at the edge of the network.

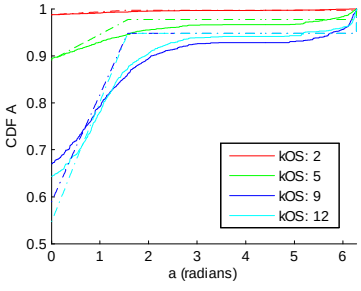
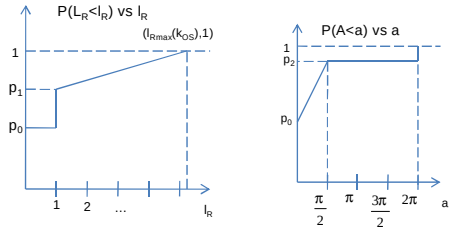


Figure 6: $P_{A|K_{OS}}(a|\,k_{OS})$ vs. a. Solid line: Computed values. Dashed lines: Model.



(a) (b)

Figure 7: (a) Idealized plot of the $P_{L_R|K_{OS}}(l_R|\,k_{OS})$. (b) Idealized plot of the $P_{A|K_{OS}}(a|\,k_{OS})$.

Figure 5 indicates that the distribution of L_R can be modeled by a simple piece-wise linear function as shown in Figure 7(a). This model takes the form

$$P_{L_R|K_{OS}}(l_R|\,k_{OS}) \approx \begin{cases} p_0(k_{OS}), & \text{if } l_R \leq 1 \\ m_R(k_{OS})(l_R - 1) + p_1(k_{OS}), & \text{if } l_R > 1 \end{cases}, \quad (13)$$

where $m_R(k_{OS})$ is the slope of the $P_{L_R|K_{OS}}(l_R|\,k_{OS})$ for $l_R > 1$ and is given by

$$m_R(k_{OS}) = \frac{1 - p_1(k_{OS})}{l_{R\max}(k_{OS}) - 1}.$$

Thus, the model of $P_{L_R|K_{OS}}(l_R|\,k_{OS})$ is characterized by parameters $p_0(k_{OS})$ and $p_1(k_{OS})$. Note that these parameters depend on the node density, however, to reduce clutter, the dependence on Δ is suppressed.

For $l_R \leq 1$, $P(L_R < l_R|\,k_{OS}) = p_0(k_{OS})$ is the probability that no nodes have paths to $\mathcal{D}$ stored in cache. In order for nodes to have paths to $\mathcal{D}$ stored in cache, $\mathcal{D}$ must transmit the $RREQ$. This can only occur if the $RREQ$ was originated by a node that is no more than $k_{OS} - 1$ hops[3] away from $\mathcal{D}$. The probability of this event occurring is denoted by $p_{DI}(k_{OS} - 1)$. Now even if $\mathcal{D}$ propagates the $RREQ$, nodes might not learn paths to $\mathcal{D}$. This occurs because by the time that $\mathcal{D}$ propagates the $RREQ$, all neighboring nodes have already received a $RREQ$ from some other node, and hence, upon checking the sequence number of the $RREQ$ transmitted by $\mathcal{D}$, all neighbors discard the $RREQ$ and no node learns about a path to $\mathcal{D}$. Denote by $\hat{p}_0$ the probability no node learns a path to $\mathcal{D}$ given that $\mathcal{D}$ propagates the $RREQ$. Therefore,

$$\begin{aligned} p_0(k_{OS}) &= (\hat{p}_0 p_{DI}(k_{OS} - 1) + (1 - p_{DI}(k_{OS} - 1))) \quad (14) \\ &= 1 - (1 - \hat{p}_0) p_{DI}(k_{OS} - 1). \end{aligned}$$

Assuming that $\mathcal{D}$ is randomly located in the $2l_X \times 2l_Y$ region and that $\mathcal{OS}$ is at the center of this region, then

[3] If it is $k_{OS} - 1$ away, then the $RREQ$ will reach D and D will propagate the $RREQ$.

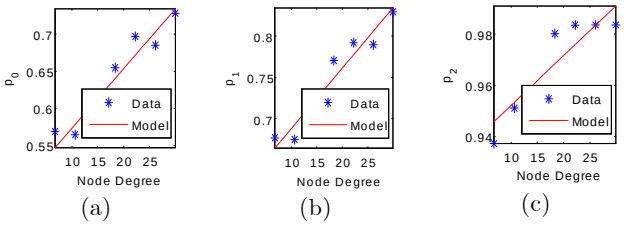


Figure 8: (a) $\hat{p}_0(\Delta)$ vs. Δ. (b) $\hat{p}_1(\Delta)$ vs. Δ. (c) $\hat{p}_2(\Delta)$ vs. Δ. stars: Computed values of parameter. solid line: parameter approximation.

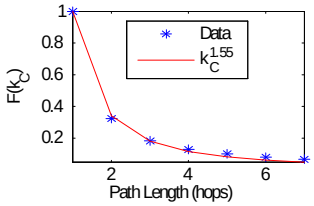


Figure 9: Scaling Factor vs Path Length. stars: Computed values of Scaling Factor. solid line: Scaling Factor approximation.

$p_{DI}(k_{OS}-1)$ is the fraction of the $2l_X \times 2l_Y$ area that is covered by a circle of radius $D(k_{OS}-1)$ centered at the center of the space. As an approximation, we assume that $p_{DI}(k_{OS}-1)$ is this fraction even if $\mathcal{OS}$ is not at the center of the region. By using (11), it is straightforward to compute p_{DI}.

The value of $\hat{p}_0$ was found from simulations and is shown in Figure 8(a). As indicated, this probability depends on the node density. For example, as the density gets larger, there are more neighbors that compete with the *RREQ* forwarded by $\mathcal{D}$, which lower the probability that nodes will learn paths to $\mathcal{D}$. Assuming that $\hat{p}_0$ is an affine function of Δ, and by minimizing the square error, we found that $\hat{p}_0 \approx \min\{1, 0.0081\Delta + 0.4930\}$.

Now we follow a similar approach to model p_1, the other parameter in (13). Specifically, for $l_R = 1$, $P(L_R < l_R | k_{OS}) = p_1(k_{OS})$, is the probability that the paths to $\mathcal{D}$ that are stored in cache are no longer than one hop. Again, this event can occur in two ways. First, it can occur because the *RREQ* was not originated close enough to $\mathcal{D}$, which occurs with probability $1 - p_{DI}(k_{OS}-2)$. Second, this event can occur because when the *RREQ* reached $\mathcal{D}$ and $\mathcal{D}$ forwarded the *RREQ* with a $TTL \geq 1$, its neighbors either discarded the *RREQ* or forwarded it, but this forwarded *RREQ* was discarded. Denote by $\hat{p}_1$ the probability no node learns a two hop path to $\mathcal{D}$ given that $\mathcal{D}$ propagates the *RREQ* with a $TTL \geq 1$. Thus, $p_1(k_{OS}) = 1 - (1-\hat{p}_1)\, p_{DI}(k_{OS}-2)$, where $\hat{p}_1$ is modeled with the affine function $\hat{p}_1 \approx \min\{1, 0.0073\Delta + 0.6155\}$. The calculated $\hat{p}_1$ and the modeled of $\hat{p}_1$ is shown in Figure 8(b).

6.2 Modeling the aperture, A

From Figure 6, we observe that the aperture can be modeled by a piece-wise linear function shown in Figure 7(b). This function takes the form

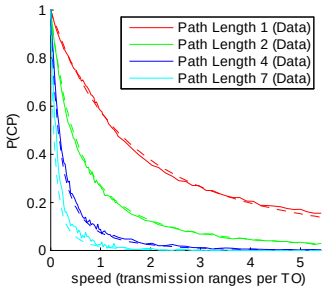


Figure 10: Proability that a path is correct $p(CP)$ vs s with $N_{OS} = 10$. Values (solid line) and approximations (dashed line) are shown.

$$P(A < a | k_{OS}) \approx \begin{cases} \frac{2(p_2(k_{OS}) - p_0(k_{OS}))}{\pi} a + p_0(k_{OS}), & \text{if } a \leq \frac{\pi}{2} \\ p_2(k_{OS}), & \text{if } a > \frac{\pi}{2} \end{cases}.$$

The parameter $p_0(k_{OS})$ is the probability that no node has a path to $\mathcal{D}$ in cache, and is given by (14). The parameter $1 - p_2(k_{OS})$ is the probability that $A = 2\pi$, that is all nodes in a disk centered around $\mathcal{D}$ all have paths to $\mathcal{D}$ in their cache. Although the aperture can never be exactly 2π in the scenario under study, in lower densities, the probability that A is nearly 2π is too large to neglect. Thus, $p_2(k_{OS}) = 1 - (1-\hat{p}_2)\, p_{DI}(k_{OS}-1)$,where $\hat{p}_2$ is the probability of $A \approx 2\pi$ given that the *RREQ* reaches $\mathcal{D}$. Based on simulations, we model $\hat{p}_2$ as $\hat{p}_2 \approx \min\{1, 0.0019\Delta + 0.9355\}$ .Figure 8(c) shows the calculated values of $\hat{p}_2$ and its model.

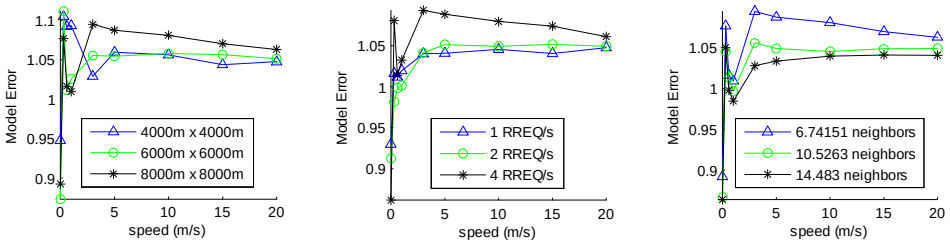
7. COMPUTING G

As mentioned in Section 3, $G(k_S, k_{OS}, d_S) := P(K_S \leq k_S | K_{OS} = k_{OS}, D_S = d_S)$ plays an important part in computing the distribution of K_S. With the models found in Sections 4-6, it is straightforward to compute G.

In this section, a method to obtain the matrix G is described. Let $p_C(k_S, k_C, d_S, l_R, a)$ be the probability of event C occurring, with $C = \{\mathcal{S}$ finds information about $\mathcal{D}$ in k_S or less hops and the length of path found is k_C, given that $\mathcal{S}$ is at a distance d_S from $\mathcal{D}$ and the $\mathcal{OS}$ made a route search that generated a pie slice with parameters l_R and $a\}$. Let $G_{K_C}(k_S, k_C, k_{OS}, d_S) := P(K_S \leq k_S, K_C = k_C | K_{OS} = k_{OS}, D_S = d_S)$, given by:

$$\begin{aligned} &G_{K_C}(k_S, k_C, k_{OS}, d_S) = \\ &= \int_0^\infty \int_0^{2\pi} P(K_S \leq k_S, K_C = k_C | K_{OS} = k_{OS}, \\ &D_S = d_S, L_R = l_R, A = a) \times \\ &p_{L_R,A}(l_R, a | K_{OS} = k_{OS}, DS = d_S)\, da\, dl_R \\ &= \int_0^\infty \int_0^{2\pi} P(K_S \leq k_S, K_C = k_C | \\ &D_S = d_S, L_R = l_R, A = a) \times p_{L_R,A}(l_R, a | K_{OS} = k_{OS})\, da\, dl_R \\ &\approx \int_{l_R} P(K_S \leq k_S, K_C = k_C | D_S = d_S, L_R = l_R, A = a) \\ &\times p_A(a | k_{OS})\, p_{L_R}(l_R, | k_{OS})\, da\, dl_R, \end{aligned}$$

where we use the fact that once L_R and A are known, K_S and K_C are independent of K_{OS}, and the fact that



(a) (b) (c)

Figure 11: Model Error vs. Speed. (a) Various Network sizes, $\Delta = 6.74$, 5 $RREQ$/s in the network. (b) Various Request Rates, $\Delta = 6.74$, Network Size is 8000x8000. (c) Various node degrees, Network Size is 8000x8000, 5 $RREQ$/s in the network.

L_R and A are independent of D_S. Furthermore, we assume that $p_{L_R,A}(l_R, a|\, K_{OS} = k_{OS}) \approx p_A(a|\, K_{OS} = k_{OS})\, p_{L_R}(l_R, |\, K_{OS} = k_{OS})$. While L_R and A are not independent, we have found that including the dependence between L_R and A does not significantly improve the quality of the model. A deeper analysis of this behavior is not included due to lack of space.

The probability $P(K_S \le k_S, K_C = k_C|\ D_S = d_S,\ L_R = l_R,\ A = a)$ is found by determining the probability that a circle of radius $D(k_S)$ centered at $(d_S, 0)$ intersects a pie-slice that has its apex at the origin and has radial length l_R and aperture a and that this intersection occurs at a distance $D(k_c)$ from the origin. Such an intersection is shown in Figure 1.

If $d_S - l_R \ge D(k_S)$ then the search area cannot intersect the pie-slice. If $d_S - l_R < D(k_S)$ but $D(k_S) < d_S$, then the search area intersects the pie-slice, only if the pie-slice is oriented between $\left[-\gamma - \frac{a}{2}, \gamma + \frac{a}{2}\right]$ where

$$\gamma = \arccos\left(\frac{r_{INT}^2 + d_S^2 - D^2(k_S)}{2 d_S \times r_{INT}}\right)$$

and $r_{INT} = \min\left\{l_R, \sqrt{\left(d_S^2 - D^2(k_S)\right)^+}\right\}$. Since the orientation of the pie-slice is uniformly distributed between $[0, 2\pi]$, the probability of the search area intersecting the pie-slice when $d_S - l_R < D(k_S) < d_S$ is $\min\{1, (2\gamma + a)/(2\pi)\}$. Finally, if $D(k_S) > d_S$, then the search area will include node $\mathcal{D}$ and hence any pie-slice will intersect the search area. Thus, the probability that the search area intersects the pie-slice is

$$p_{INT}(k_S, d_S, l_R, a) = \begin{cases} 0, \text{ if } d_S - l_R \le D(k_S) \\ \min\left\{1, \frac{2\gamma + a}{2\pi}\right\} \text{ if } d_S - l_R < D(k_S) < d_S \\ 1 \text{ otherwise} \end{cases}. \quad (15)$$

Given K_{OS}, D_S, L_R and A and that the orientation and aperture is such that an intersection occurs, K_C is exactly known. Specifically,

$$\bar{k}_C(k_S, d_S, l_R, a) = round(L(r_{INT})). \quad (16)$$

Therefore, using 15 and 16, $p_C(k_S, k_C, d_S, l_R, a)$ is given by:

$$P(K_S \le k_S, K_C = k_C|\, D_S = d_S, L_R = l_R, A = a) = \begin{cases} p_{INT}(k_S, d_S, l_R, a), \text{ if } k_C = \bar{k}_C(k_S, d_S, l_R, a) \\ 0, \text{ otherwise} \end{cases}$$

Finally, G can be found by summing over all possible values of K_C, i.e., $G(k_S, k_{OS}, d_S) = \sum_{k_C} G_{K_C}(k_S, k_C, k_{OS}, d_S)$.

8. SPEED MODEL

Paths stored in cache might break as a consequence of node mobility. Clearly, longer paths have a higher probability of breaking than shorter paths. Thus, the probability that a route search returns a broken path depends on K_C, the length of the path retrieved from an intermediate node's cache. Let $P(CP|\, k_C; s, N_{OS})$ be the probability that a path in cache is correct, given it's length is k_C, there are N_{OS} old sources in the system, and nodes are moving at a speed s. Figure 10 shows values of $P(CP|\, k_C; s, N_{OS})$ found from simulations. The upper line corresponds to the path of length one, i.e., $P(CP|\, 1; s, N_{OS})$. In this section, it will be shown that $P(CP|\, k_c; s, N_{OS})$ can be computed from $P(CP|\, 1; s, 1)$, which is approximated with a hyperexponential distribution.

While it is not the case, it is informative to consider the simple case where link lifetimes are independent and exponentially distributed. In this case, $P(CP|\, 1; s, N_{OS}) = \exp(-\lambda/s)$, for some λ that depends on the cache timeout. Moreover, in this case, $P(CP|\, k_c; s, N_{OS}) = \exp(-\lambda/(s \times k_C))$ $= P(CP|\, 1; s \times k_C, N_{OS})$, Thus, the probably of a path of length k_C breaking can be found from the probability of a path of length one breaking, but with a different speed. While the independence and exponential distributed link lifetime provides a low quality fit, simulations indicate that a similar scaling does hold. Specifically, simulations indicate that

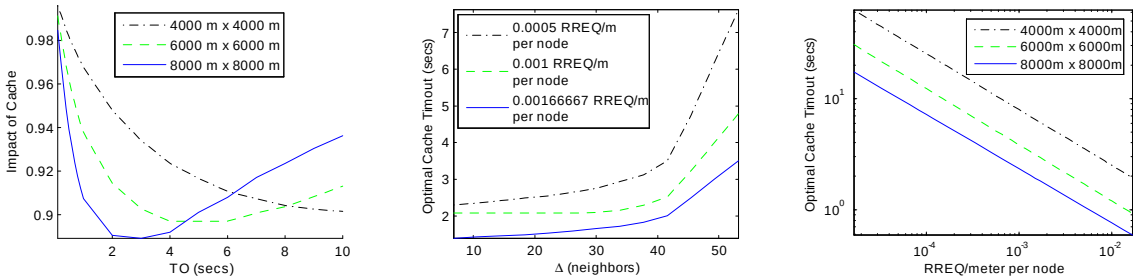
$$P(CP|\, k_C; s, N_{OS}) \approx P(CP|\, 1; s \times F(k_C), N_{OS}).$$

The function $F(k_C)$ can be found as follows. Through simulations, $P(CP|\, k_C; s, N_{OS})$ can be estimated for a fixed values of N_{OS} and different values k_C and s. Then, for each value of k_C we compute

$$F(k_C) = \arg\min_F \int_0^{10} |P(CP|\, k_C; s, N_{OS}) - P(CP|\, 1; s \times F, N_{OS})|\, ds,$$

where speed is measured in transmission ranges per cache timeout. Figure 9 shows $F(k_C)$ for several different values of k_C. As can be observed, $F(k_C) \approx k_C^{1.55}$.

Now we consider the impact of the number of old sources N_{OS}, on the probability that a path in cache is stale. Note that when we say that there were N_{OS} previous searches, we mean that in the past TO seconds, there have been N_{OS} route searches, where TO is the cache timeout (i.e., routes in cache are deleted after TO seconds). Here we assume that the N_{OS} previous searches are periodically distributed over the past TO seconds. Thus, the most recent search occurred TO/N_{OS} seconds before the current search, next to most recent search occurred $2 \times TO/N_{OS}$ seconds before



(a) (b) (c)

Figure 12: (a) Impact of Cache vs Cache Timeout for 3 different network sizes. $\Delta = 6.74$,$\nu = 5 \times 10^{-4}$ $RREQ/m$ per node.(b) Optimal Cache Timeout vs. Δ for different request rate/speed ratios.Network Size is $8000m \times 8000m$.(c) Optimal Cache Timeout vs. Request Rate/size ratio for different sizes. Average node degree $\Delta = 6.74$.

the current search, and so on.

Note that speed and the cache timeout are interchangeable in the following sense. If a node moves at speed s, then the distance it moves over time t is the same as it would move over a time $t \times M$ if its speed were s/M. Letting $P_{TO}\left(CP|\,1;s,1\right)$ denote the probability that a path stored in cache is broken before the path is deleted due to cache timeout, when the cache timeout is TO. Then, $P_{TO/M}\left(CP|\,1;s,1\right) \approx P_{TO}\left(CP|\,1;s\times M,1\right)$. Therefore, the probability that the path stored in cache from the most recent route search is is not stale is $P_{TO}\left(CP|\,1;s\frac{1}{N_{OS}},1\right)$. In general, the probability that paths saved into cache from the n-th most recent route search remain valid is $P_{TO}\left(CP|\,1;s\frac{n}{N_{OS}},1\right)$.

When a route search returns a path that was stored in cache, this path was saved into cache during one of the past N_{OS} route searches. The probability that the route returned was saved into cache during a particular route search is $1/N_{OS}$. Thus

$$p\left(CP|\,1;s,N_{OS}\right)=\frac{1}{N_{OS}}\sum_{n=1}^{N_{OS}}p\left(CP|\,1;s\frac{n}{N_{OS}},1\right).$$

Hence, in order to compute $p\left(CP|\,k_C;s,N_{OS}\right)$, we only need to know $p\left(CP|\,1;s,1\right)$. As mentioned above, assuming that link lifetimes are exponentially distributed provide a low quality fit. However, a hyperexponential distribution does provide a high quality fit. Specifically, we use $p\left(CP|\,1;s,1\right)\approx a_3\exp\left(-a_1 s\right)+\left(1-a_3\right)\exp\left(-a_2 s\right)$, where $a_1 = 15.1$, $a_2 = 7.4$, $a_3 = 0.5$. Therefore, the complete model is then given by:

$$p\left(CP|\,k_C;s,N_{OS}\right)=\frac{1}{N_{OS}}\sum_{n=1}^{N_{OS}}a_3\exp\left(-a_1 s\frac{n}{N_{OS}}k_C^{A_S}\right)+\left(1-a_3\right)\exp\left(-a_2 s\frac{n}{N_{OS}}k_C^{A_S}\right).$$

Figure 10 show the quality of fit for the $N_{OS} = 10$ case.

9. MODEL VALIDATION

In this section the model is compared against data obtained by simulation. To this end, simulations were run varying each one of the parameters that affect the behavior of the system, namely, node speed, node density (node degree), route request rate, and network size. By running 50000 trials, the value of $E\left(K_S\right)$ is estimated for each set of system parameters investigated, where K_S is the number of hops a route search message must travel before reaching a node with a path to the destination in cache, or reaching the destination itself. This estimate is denoted by $\hat{E}\left(K_S\right)$. In order to determine the quality of the cache model, the model is used to estimate $E\left(K_S\right)$, i.e., $E\left(K_S\right)\approx\sum_{k_s=0}^{\infty}k_s p_{K_S}\left(k_s\right)$. Specifically, Figure 11 (a), (b), and (c) show the relative error $\left(\hat{E}\left(K_s\right)-\sum_{k_s=0}^{\infty}k_s p_{K_S}\left(k_s\right)\right)/\hat{E}\left(K_S\right)$.As can be observed, in almost all cases the model can give an estimate within 10% error.

10. PERFORMANCE OF TOPOLOGY CACHE

The model of topology cache allows the performance of cache to be analyzed over a wide range of system and environmental parameters. In this section, a few aspects of performance are analyzed, namely, the optimal cache timeout, which has been the subject of prior simulation-based research, and the performance improvement provided by using cache.

10.1 Optimal Cache Timeout

In this section a optimal cache timeout TO^* is investigated, where $TO^* = \arg\min_{TO} E\left(K_S\right)$. For this study it is important to note that the variables cache timeout TO, speed s, and request rate per node λ are related in the following manner: If a particular network $\mathcal{A}$ with nodes generating λ_A $RREQ$/sec and nodes moving at s_A m/sec has an optimal $TO^* = t_A$, another network $\mathcal{B}$ with parameters $\lambda_B = 2\lambda_A$ and $s_B = 2s_A$ will have the same optimal cache timeout, i.e., $t_B = t_A$. Therefore the parameters request rate and speed can be merged into a single parameter $\nu = \lambda/s$, which is the number of route request a node generates when it moves a meter.

Denote by $\eta\left(TO\right)$ the ratio between $E\left(K_S\right)$ obtained when using the cache with timeout set to TO and $E\left(K_S\right)$ obtained when no cache is used. A value of $\eta\left(TO\right) = 1$ indicates that utilizing cache does not improve performance. A lower value of $\eta\left(TO\right)$ indicates the network can benefit from the use of cache. Clearly, $TO^* = \arg\min_{TO}\eta\left(TO\right)$.

Figure 12(a) shows the values of $\eta\left(TO\right)$ for three different scenarios. Note that $\eta\left(TO\right)$ is more sensitive for $TO < TO^*$ than when $TO > TO^*$. Thus, we conclude that it is preferable to overestimate the cache timeout than to underestimates it. Figure 12(a) also shows relatively low sensitivity for $TO \approx TO^*$ in the sense that $\eta\left(TO\right)$ is nearly unchanged

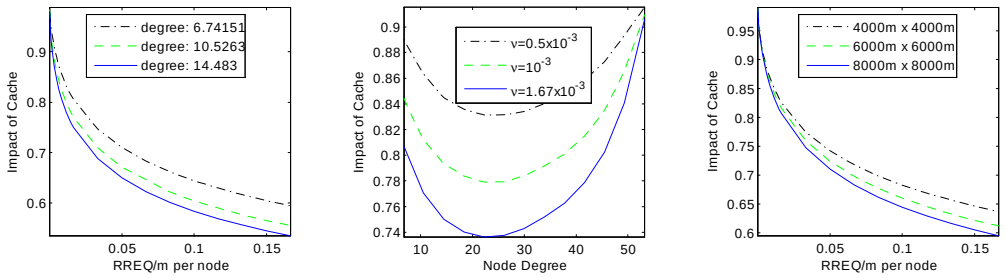


Figure 13: (a) Impact of cache vs. Request Rate/speed ratio for different node degrees. Network Size is 8000m x 8000m. (b) Impact of cache vs. Node Degree for different Request Rate/speed ratios. Network Size is 8000m x 8000m. (c) Impact of cache vs. Request Rate/speed ratio for different network sizes. Average node degree $\Delta = 6.74$.

if $TO \in [TO^* - 1, TO^* + 1]$.

Figure 12(b) shows TO^* as a function of Δ for three values of ν. Observed that for $\Delta \leq 40$ *neighbors* and ν held constant, the value of TO^* remains nearly constant (recall that ± 1 sec has little or no impact on performance). Consequently, once TO^* has been determined based on the network size, the rate of $RREQ$, and node speed, TO does not need to be changed as nodes enter or leave the network. Note that the insensitivity of TO^* to Δ is not equivalently to insensitively of $\eta(TO^*)$ to Δ. For example, Figs. 13(a) and 13(b) show that $\eta(TO^*)$ does change with respect to density. Further investigation is required to understand the reasons for this insensitivity of TO^* to changes in Δ.

Figure 12(c) shows the values of TO^* as a function of ν for different network sizes. A polynomial relationship between TO^* and ν is observed, i.e., $TO^* \approx A \times \nu^B$. Furthermore, the parameter B is independent of size of the network.

10.2 Impact of Cache

Figures 13(a), (b) and (c) illustrate the impact of topology cache for a wide range of environmental parameters. In all cases, the optimal cache timeout is used. Figure 13(b) shows how the impact of cache varies with the node density. The node density impacts performance of cache in two ways. When the node density is very high and a node forwards a $RREQ$, there are a large number of other neighboring nodes that will also forward the $RREQ$. But a node that receives the $RREQ$s will discard all but the first $RREQ$ received. Thus, when the node density is high, most $RREQ$s are discarded. In this way, increasing the node density acts to reduce the impact of cache. On the other hand, since the rate that a node generates route searches is fixed, as the node density increases, the rate of route searches are generated throughout the network increases. In this way, increasing the node density acts to increase the impact of cache. Figure 13(b) shows that this conflicting impact of node density initially increase the utility of cache, but later decrease the utility of cache.

Figures 13(a) and 13(c) illustrate the impact of cache as a function of ν. As expected, ν increases the utility of cache. Figure 13(c), as the size of the network increases, the utility of cache also increases. Note that Fig.13(c) shows that the cache reduces the number of hops a $RREQ$ must travel by 40%.

11. CONCLUSIONS

This paper presents a model of DSR-style topology cache. The model describes the steady-state performance of the routing protocol by finding the fixed point of the cache dynamics. With this model optimal cache timeout can be easily computed. It also allows an accurate comparison between DSR and other protocols such as OLSR. In future work, the model developed here will be extended to include alternative flooding schemes such as the ones described in [3], as well as to include heterogeneous propagation such as urban propagation.

Disclaimer

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U. S. Government.

12. REFERENCES

[1] D. Johnson, Y. Hu, and D. Maltz, "The dynamic source routing protocol (dsr) for mobile ad hoc networks for ipv4," RFC 4728 (Experimental), Internet Engineering Task Force, Feb. 2007.

[2] R. J. La and Y. Han, "Distribution of path durations in mobile ad hoc networks and path selection," IEEE/ACM Trans. Netw., vol. 15, no. 5, pp. 993–1006, 2007.

[3] B. Williams and T. Camp, "Comparison of broadcasting techniques for mobile ad hoc networks," in MobiHoc '02. ACM, 2002, pp. 194–205.

[4] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in MobiCom '98. ACM, 1998, pp. 85–97.

[5] W. Navidi and T. Camp, "Stationary distributions for the random waypoint mobility model," IEEE Transactions on Mobile Computing, vol. 3, no. 1, pp. 99–108, 2004.