A compact routing protocol for ad hoc networks

Minghua Chen
Dept. of Electrical Engineering and Computer Sciences
University of California, Berkeley, CA 94720, USA
minghua@eecs.berkeley.edu

Ayalvadi Ganesh
Microsoft Research
7 J J Thomson Ave, Cambridge CB3 0FB, UK
ajg@microsoft.com

***Abstract*— We present a scheme for routing between arbitrary source and destination nodes in a wireless network based on node identifiers. Nodes are aware of their own location, but not of the location of all other nodes. Our scheme provides a distributed name to location directory. It achieves constant stretch with polylogarithmic storage at each node. Join and leave operations incur polylogarithmic cost.**

I. INTRODUCTION

In recent years, there has been significant research interest in wireless ad hoc, mesh and sensor networks. An important goal is to provide communication functionality between nodes without relying on any external infrastructure. The network is typically too large to allow direct one-hop communication between all node pairs. Hence nodes have to cooperate to route messages from source to destination.

Ad hoc routing has been the topic of much previous work. Currently, two of the best known schemes are AODV (ad hoc on demand distance vector routing) [14] and DSR (dynamic source routing) [10]. Route discovery in DSR takes place by broadcasting the route request to all nodes in the network; the cost of such an operation, measured by the number of transmissions required, is of the order of n, where n is the total number of nodes in the network. Moreover, after a route has been discovered, the source node has to store the entire route and include it in every transmitted packet; if the n nodes are located on a plane, this can require of the order of $\sqrt{n}$ intermediate node identifiers. AODV also uses a broadcast mechanism to discover routes. The main difference is that routing information is maintained on intermediate nodes rather than by just the source node, and hence doesn't need to be included in each packet header. The trade-off is that each node has to maintain next hop information for every destination to which it is an intermediary; this requires $O(n)$ storage in the worst case but is expected to be much less in practice as only a small fraction of nodes are typically active simultaneously. In the case of route failure, for example due to node mobility, both AODV and DSR re-initiate route discovery in order to repair the route. As this involves broadcast, it is potentially expensive. AODV and DSR are reactive, and discover routes in response to a request. Schemes such as OLSR (optimized link state routing) [9] maintain routes pro-actively, with the aim of reducing message latency, but have similar memory requirements to AODV and DSR.

The approach taken in this paper is different. We assume that nodes know their own location in some co-ordinate system. This could be achieved if nodes carry GPS receivers, or using any of several distributed schemes that have been proposed; see [12] and references therein, for example. If the source of a message knows the location of the destination, the message can be forwarded using a greedy routing scheme described in Section II, or any other scheme proposed in the literature, such as [11]. In this case, the packet header only needs to carry the ID and location of the destination. However, for every source to know the location of every destination requires $\Omega(n)$ storage per node; in addition, it requires changes such as node join, leave or move to be broadcast throughout the network. The main contribution of our work is a distributed node ID to location mapping (directory) which can be used in conjunction with any scheme for nodes to find their own location, and to communicate between known locations.

There is a fundamental trade-off between the *space* used to store the routing table at each node and the *stretch* of the routing scheme, namely the worst-case ratio of the length of the routing path used by the scheme to the length of the shortest path. There is a large corpus of work studying this trade-off; see [7] for a comprehensive survey. There are two main variants of the routing problem: the *name-independent* case, where node names are given, and the *labelled* case, where the algorithm can assign pre-computed labels (of small size, say polylogarithmic in n) to nodes; labels may contain, e.g., location information. The latter are only relevant to static networks and are not of interest to us. Peleg and Upfal [13] showed that any routing scheme, even a labelled one, achieving stretch k on arbitrary networks requires a total of $\Omega(n^{1+\frac{1}{2k+4}})$ bits of storage over all nodes. Thorup and Zwick [16] presented a labelled scheme which achieves stretch $2k-1$ using $\tilde{O}(n^{1/k})$ bits per node. (We write $f(n) = \tilde{O}(g(n))$ to mean that $f(n)/g(n) = O(\log^{\beta} n)$ for some β.) Name-independent schemes achieving stretch 5 and 3 respectively with $\tilde{O}(\sqrt{n})$ bits per node were presented in [4] and [1]. Better results are known for networks with special structure. For example, it is shown in [6], [16] that labelled routing with stretch 1 can be achieved on trees using $O(\log^2 n/\log\log n)$ bits of storage per node. For name-independent routing, Abraham and Malkhi [2] showed how to achieve constant stretch (> 1) on growth bounded networks using $O(\log^5 n)$ bits of storage per node, while Abraham *et al.* [3] presented schemes achieving constant stretch on networks with doubling dimension α using $2^{O(\alpha)} \log^4 n$ bits per node. (A network is called growth bounded if the number of nodes within distance $2r$ of any node is bounded by a constant times the number within distance r, for any r. A network is said to have doubling dimension α if any ball of radius $2r$ can be covered by at most 2^{α} balls of radius r. A ball with centre u and radius r is defined as the set of all nodes within distance r of u; the centre need not be unique.)

In this paper, we make the much stronger assumption that nodes are uniformly distributed on a square. We then present a routing scheme that achieves constant stretch (with high probability) and requires each node to store location information for $O(\log^2 n)$ other nodes. The main advantage of the scheme is its simplicity and the ease of making incremental changes, such as to deal with node join and leave. Hence, it is also well suited for applications where node mobility is an issue. We remark that while [2], [3] solve the same problem in a more general setting, the schemes they present are not easily amenable to a distributed implementation. The routing tables are constructed using a fairly complex global scheme and can then support simple routing decisions. While this is acceptable for static networks, it is not well suited for networks where nodes may join and leave frequently, or where mobility is an issue.

The paper unfolds as follows. We present the system model and describe our protocol in Section II, including stretch and storage requirement analysis. NS-2 simulation results are presented in Section III. Section IV concludes the paper.

II. System model and protocol description

Suppose for simplicity that there are exactly 4^n nodes on a square of side 2^n, so the average node density is 1. Suppose nodes are assigned distinct IDs uniformly at random; this can be achieved by hashing node IDs, for example. By a node u, we will mean the node with ID $u = u_1 u_2 \cdots u_n$ in base 4. The location of node u is denoted $x(u)$. The routing table of node u consists of the locations of a subset of other nodes. We now describe how to populate routing tables, i.e., how to choose nodes whose locations are stored.

We use r to denote a constant (which may depend on n but not on the node ID or location) which will be specified later. Node u maintains IDs of:

- All nodes within distance $2r$ as its Level 0 routing table entries;
- All nodes with ID u_1* within distance $4r$ as its Level 1 routing table entries; and likewise,
- All nodes with ID $u_1 u_2 \cdots u_k*$ within distance $2^{k+1}r$ as its Level k routing table entries, for each $k = 0, \ldots, n$.

As usual, u_1* refers to all IDs with first digit u_1 and so on.

Let $B(u, r)$ denote the ball of radius r centred at u. Clearly, the expected number of Level k routing table entries is no more than $4\pi r^2$; to see this, note that the density of nodes with prefix $u_1 u_2 \ldots u_k$ is 4^{-k}, and the area of the intersection of $B(u, 2^{k+1})$ with the square on which the nodes are located is at most $\pi 4^{k+1} r^2$. Summing over k, the expected size of node u's routing table is $4\pi r^2(n+1)$. Later, we shall see that r^2 is chosen to be proportional to n. Therefore, the number of routing table entries per node is quadratic in the logarithm of the number of nodes.

Before specifying the routing algorithm, we state our assumptions about the communication model. We assume that all nodes have the same finite range and communicate over point-to-point links. We take the range to be $2r$, which can be ensured by choosing the transmit power appropriately, if we suppose that the transmission range is limited by noise rather than interference. (If this is not the case, a more complex scheme involving time or code-sharing will be needed to achieve a range of $2r$. We don't discuss the details as this is not the focus of our work.) Thus, we can represent the system as a graph in which there is an edge between any two nodes within distance $2r$ of each other. We denote this graph $G(4^n, 2r)$. Any communication between nodes more than distance $2r$ apart has to be routed via intermediate nodes.

We have used the same constant r in specifying both the routing table construction and the transmission range in order not to overburden the notation. In fact, our routing protocol description is unchanged if the transmission range is $2R$ and we choose $R \geq r$; if $R < r$, it needs a somewhat more complicated implementation (and may be impossible to implement if R is too small).

The routing protocol that we now describe comprises two distinct algorithms: (i) *Algorithm A* aims to find a physical location corresponding to a given node ID, while (ii) *Algorithm B* routes between nodes with known locations via intermediate nodes, if necessary.

A. Algorithm A

Suppose node u wants to route a message to node $v = v_1 v_2 \cdots v_n$. It does along a path of intermediate nodes $w^1, w^2, \ldots$ specified as follows.

- If $v \in B(u, 2r)$, u routes directly to v; otherwise, it routes to an intermediate node $w^1 = v_1* \in B(u, r)$, assuming for now that such a node exists.
- If $v \in B(w^1, 4r)$, then w^1 knows this by construction of its routing table, and routes directly to v. Otherwise, it routes to a node $w^2 = v_1 v_2* \in B(w^1, 2r)$.
- Likewise, the intermediate node $w^k = v_1 v_2 \cdots v_k*$ routes directly to v if $v \in B(w^k, 2^{k+1}r)$ and to $w^{k+1} = v_1 v_2 \cdots v_{k+1}* \in B(w^k, 2^k r)$ otherwise.

Note that if suitable intermediate nodes can be found at each step of the routing algorithm, then it is a form of prefix routing and is guaranteed to reach the destination v in at most n steps. We shall later address how r should be chosen so that such intermediate nodes exist at each step of the routing protocol with high probability (whp), i.e., with probability going to 1 as $n \to \infty$. For definiteness, we need to specify what to do in the unlikely event that such a node can't be found. We assume in this case that the routing protocol resorts to flooding the network, like in AODV. Better choices may be possible, but this will be a sufficiently rare event as to not affect average performance measures.

B. Algorithm B

Suppose nodes u and v with known locations $x(u)$ and $x(v)$ are further than $2r$ apart. In order to specify how a message is routed from u to v, we need some definitions. We say that the graph $G(4^n, 2r)$ is θ-connected if, for any node u and any cone of angle θ with vertex at u, there is at least one other node within distance $2r$ of u. (In fact, we will not impose this requirement for nodes which are within distance $2r$ of an edge of the square, but we ignore edge effects for ease of exposition.) This definition is motivated by, and closely related to, the concept of θ-coverage introduced by Xue and Kumar [17], but not identical to it. Note that $G(4^n, 2r)$ is connected if it is 2π connected, and that the smaller the value of θ, the more stringent is the requirement that it be θ-connected. We shall henceforth assume that r has been chosen large enough that $G(4^n, 2r)$ is $\pi/3$-connected. We will discuss how to choose r in a later subsection.

Node u routes the message to v through a sequence of intermediate nodes w^i with $w^0 = u$. Each intermediate node w^j routes the message to one of its neighbors w^{j+1} which is in the sector of angle $\pi/3$ centred on the ray (w^j, v); it does not matter which but, for definiteness, we can assume that it chooses the node which is closest to v. Since the distance to v strictly decreases after each routing step, the message is guaranteed to reach v after at most 4^n routing hops.

C. Analysis of stretch

We first consider Algorithm B, routing a message from node u to node v, whose coordinates are known. Let $w^1, w^2, \ldots$ denote the sequence of intermediate nodes through which the message is routed. Let $D_j = d(w^j, v)$ denote the Euclidean distance between w^j and v, and let $d_{j+1} = d(w^j, w^{j+1})$. Let θ_{j+1} denote the angle between the rays (w^j, v) and (w^j, w^{j+1}) and note that $\theta_{j+1} \leq \pi/6$ by the specification of the routing algorithm. Using the well-known trigonometric identity

$$\begin{aligned} D_{j+1}^2 &= D_j^2 + d_{j+1}^2 - 2D_j d_{j+1}\cos(\theta_{j+1}) \\ &\leq D_j^2 + d_{j+1}^2 - 2D_j d_{j+1}\cos\frac{\pi}{6}, \end{aligned}$$

we find that $\Delta_{j+1} := D_{j+1} - D_j$ satisfies

$$\frac{\Delta_{j+1}}{d_{j+1}} \leq \frac{\sqrt{3}D_j - d_{j+1}}{D_{j+1} + D_j} < \frac{(\sqrt{3}-1)D_j}{2D_j}. \qquad (1)$$

To obtain the last inequality, we have used the fact that $D_{j+1} < D_j$ (the distance to v strictly decreases with each routing step) and that $d_{j+1} \leq D_j$ (otherwise, v is within range of w^j and w^j would route directly to v). In words, (1) says that each routing step reduces the Euclidean distance to the target by at least $(\sqrt{3}-1)/2$ times the length of the step. Consequently, the total distance traversed by the message is at most $2/(\sqrt{3}-1)$ times the Euclidean distance between u and v. Therefore, the stretch due to Algorithm B is bounded above by $2/(\sqrt{3}-1)$, which in turn is smaller than 3.

Next, we consider Algorithm A which seeks to route a message from a node u to another node v whose coordinates may not be known to u. Let $w^1, w^2, \ldots$ be the sequence of intermediate nodes through which the message is routed. Observe that $w^k \in B(w^{k-1}, 2^{k-1}r)$ unless $w^k = v$; hence, by induction, $w^k \in B(u, (2^k-1)r)$, for each intermediate node. Therefore, $B(w^k, 2^{k+1}r) \supset B(u, 2^k r)$. In particular, if $v \in B(u, 2^k r)$, then $w^{k+1} = v$. More precisely, it can be seen that if k is the smallest whole number for which $v \in B(u, 2^k r)$, then the smallest j for which $w^j = v$ is either k or $k+1$.

We now compute the total distance travelled by a message from u to v. Suppose $w^k \neq v$ but $w^{k+1} = v$. Now, defining $w^0 = u$, the sum of Euclidean distances along the path from u to w^{k-1} satisfies

$$\sum_{j=1}^{k-1} d(w^{j-1}, w^j) \leq \sum_{j=1}^{k-1} 2^{j-1}r = (2^{k-1}-1)r.$$

Since $w_k \neq v$, it must be the case that $d(w^{k-1}, v) > 2^k r$. But $d(u, w^{k-1}) \leq (2^{k-1}-1)r$ as noted above. So $d(u,v) > (2^{k-1}+1)r$ by the triangle inequality. Also,

$$\begin{aligned} \sum_{j=1}^{k} d(w^{j-1}, w^j) + d(w^k, v) &\leq (2^k-1)r + d(u, w^k) \\ &\quad + d(u,v) \\ &\leq 2(2^k-1)r + d(u,v). \end{aligned}$$

Since $d(u,v) > (2^{k-1}+1)r$, it follows that

$$\begin{aligned} \frac{\sum_{j=1}^{k+1} d(w^{j-1}, w^j)}{d(u,v)} &< \frac{2(2^k-1)r}{(2^{k-1}+1)r} + 1 \\ &= 1 + \frac{4 - \frac{1}{2^{k-1}}}{1 + \frac{1}{2^{k-1}}} < 5. \end{aligned} \qquad (2)$$

Combining this with the fact that the stretch of Algorithm B (which is used for routing between successive intermediate nodes w^j, w^{j+1} of Algorithm A) is at most 3, we find that the overall stretch of the routing protocol is at most 15.

Remarks: An alternative definition of stretch would be with respect to graph distance in $G(4^n, 2r)$ rather than Euclidean distance. We don't provide analytical results for this metric but evaluate it through simulation. Theoretical bounds on the stretch with respect to graph distance can be obtained using the relationship between Euclidean distance and number of hops explicated in [5]. The analysis above is not very tight, but suffices to show that constant worst-case stretch can be achieved with polylogarithmic memory per node. Simulations show that the constant is much better than 15 on average.

D. Join and Leave

The routing tables constructed by the algorithm described above have the property that, if node u is among node v's Level k routing table entries, then v must also be in Level k of u's routing table. This pairwise symmetry property is the key to making join and leave operations easy.

Before a node, say u, leaves the network, it needs to notify all nodes that have it as a routing entry. By the pairwise symmetry property, these are precisely the nodes that comprise u's routing table, so u can leave after notifying all nodes in its routing table. Thus, the number of nodes which need to be notified of a leave is $O(n^2)$ (provided $r^2 = O(n)$, as we shall show).

When node u joins the network, it needs to build up its routing table and to update those of nodes that it includes in its routing table. In order to facilitate a join protocol as decentralized as the leave protocol above, we need some additional assumptions. These are slightly stronger than the assumption of $\pi/3$-connectivity made earlier.

Assumption A We assume that, with high probability, every node u has the property that for each $k \in \{1, \dots, n\}$, every sector of the ball $B(u, 2^k r)$ subtending an angle of $2\pi/3$ at the centre contains at least one node with ID $u_1 u_2 \cdots u_k *$.

We also need the following elementary geometric fact:

Lemma 1: Suppose every sector of $B(u, r)$ subtending an angle $2\pi/3$ at the centre contains at least one other node v_i. Then $B(u, 2r) \subseteq B(v_i, 2r)$, where the union is taken over all nodes $v_i \neq u$ in $B(u, r)$.

Proof. Let $x \in B(u, 2r)$. Consider the sector of $B(u, r)$ of angle $2\pi/3$ centred on the ray (u, x); by assumption, there is at least one node v_i in this sector. We shall show that $x \in B(v_i, 2r)$, i.e., that $d(v_i, x) \leq 2r$. Denoting the angle between the rays (u, x) and (u, v_i) by θ, we have $\theta \leq \pi/3$. Now,

$$\begin{aligned} d(v_i, x)^2 &= d(v_i, u)^2 + d(x, u)^2 - 2d(v_i, u)d(x, u)\cos\theta \\ &\leq d(v_i, u)^2 + d(x, u)^2 - d(v_i, u)d(x, u) \\ &\leq \frac{3}{4}d(x, u)^2 + \Big(d(v_i, u) - \frac{1}{2}d(x, u)\Big)^2 \\ &\leq 3r^2 + \max\{r^2, r^2\} \;=\; 4r^2, \end{aligned}$$

where the last inequality follows from the fact that $d(x, u) \leq 2r$ and $d(v_i, u) \leq r$. Thus, $d(v_i, x) \leq 2r$, and the proof of the lemma is complete. □

The join protocol for a node u works as follows:

1) As a bootstrap, u discovers its one-hop neighbors by locally broadcasting a "hello" message and getting replies from all neighbors within distance $2r$. These constitute its Level 0 routing table entries.
2) Next, it uses the Level 1 routing table entries of all its Level 0 neighbors v_i with ID u_1* in order to construct its own Level 1 routing table. (In fact, it need only use enough of them for $B(v_i, 4r)$ to cover $B(u, 4r)$. This can be achieved using at most six neighbors.) By Assumption A and Lemma 1, the balls $B(v_i, 4r)$ cover $B(u, 4r)$. Hence, all nodes u_1* in $B(u, 4r)$ belong to the Level 1 routing table of some Level 0 neighbor v_i of u sharing its first digit. Finally, location information can be used to filter out those Level 1 entries of v_i's routing table that don't belong in $B(u, 4r)$.
3) Following the same idea, u can build its routing table recursively. To build Level k, u selects all nodes v_i with ID $u_1 u_2 \cdots u_k *$ in Level $k-1$ of its routing table; by definition, these nodes belong to $B(u, 2^k r)$. It then uses the Level k routing table entries of these nodes to construct its own Level k routing table. The construction is feasible and complete by Assumption A and Lemma 1.
4) After u finishes building its routing table, it sends out messages to all nodes in its routing table, asking them to add its ID and location into their own routing tables. These updates are sufficient due to the pairwise symmetry property.

In the low probability event that Assumption A fails for some node u at some Level k, it could resort to flooding (with limited range) in order to locate nodes in the uncovered region.

By the above process, u successfully adds itself into the network. The message complexity of the join operation is: one one-hop broadcast message to find Level 0 neighbors, and up to six messages at each of the n levels. Each of these messages involves nodes sending one level of their routing table to node u, and hence has size $O(n)$ in terms of number of entries. Finally, node u informs all $O(n^2)$ of its routing table entries that it has joined, which involves a message of size $O(1)$. Thus, the total cost of a join is $O(n^2)$. We do not analyse the cost in the event that Assumption A fails, as we can make its probability sufficiently small that it won't affect the average cost.

E. Choice of r

The parameter r has three roles in the routing protocol, and needs to be chosen large enough to fulfil all of them. First, $G(4^n, 2r)$ needs to be $\pi/3$-connected, in order for Algorithm B to work. (In particular, this guarantees connectivity.) Second, r should be large enough to guarantee that each ball $B(u, 2^{k-1} r)$ contains at least one node with each of the four prefixes $u_1 u_2 \cdots u_{k-1} y$, for $y = 0, 1, 2, 3$. This is needed for the prefix routing algorithm A to work. Finally, we would like r to satisfy Assumption A, so that join operations can be supported in a simple, decentralized manner. We now compute bounds on the probability of each of these conditions being violated, for a given value of r. We ignore edge effects in the analysis.

First, note that $G(4^n, 2r)$ will be $\pi/3$-connected if, for each node u, and a *fixed* partition of $B(u, 2r)$ into sectors of angle $\pi/6$, every sector contains at least one node. This is because any sector of angle $\pi/3$ in $B(u, 2r)$ must fully contain one of the fixed sectors of angle $\pi/6$. Let $\alpha(u)$ denote the probability that this requirement is violated by node u. Then,

$$\alpha(u) \leq \Big(1 - \frac{4\pi r^2}{6 \cdot 4^n}\Big)^{4^n} \leq e^{-2\pi r^2/3}. \tag{3}$$

Similarly, Assumption A is satisfied if, for each u and k, and a fixed partition of $B(u, 2^k r)$ into sectors of angle $\pi/3$, every sector contains at least one node with prefix $u_1 u_2 \cdots u_k$. Let $\alpha_k(u)$ denote the probability that this fails for a given k and u. Since the number of nodes with prefix $u_1 u_2 \cdots u_k$ is 4^{n-k}, we have

$$\alpha_k(u) \leq \Big(1 - \frac{\pi 4^k r^2}{3 \cdot 4^n}\Big)^{4^{n-k}} \leq e^{-\pi r^2/3}. \tag{4}$$

Finally, let $\beta_k(u)$ denote the probability that for some level k and node u, the ball $B(u, 2^{k-1}r)$ fails to contain any node with prefix $u_1u_2\ldots u_{k-1}y$, for some $y \in \{0,1,2,3\}$. Since there are 4^{n-k} nodes with each k-digit prefix, we have

$$\beta_k(u) \leq \left(1 - \frac{\pi 4^{k-1} r^2}{4^n}\right)^{4^{n-k}} \leq e^{-\pi r^2/4}. \qquad (5)$$

Now let γ denote the probability that our assumptions are violated at some node u. By the union bound, and (3), (4), (5), we have,

$$\begin{aligned}\gamma &\leq \sum_u \alpha(u) + \sum_{u,k} [\alpha_k(u) + \beta_k(u)] \\ &\leq 4^n \left(e^{-2\pi r^2/3} + ne^{-\pi r^2/3} + ne^{-\pi r^2/4}\right).\end{aligned}$$

Thus, for any $\delta > 0$,

$$r^2 \geq (1+\delta)\frac{4\log 4}{\pi} n \;\Rightarrow\; \gamma \leq (2n+1)4^{-\delta n}. \qquad (6)$$

As the bound above goes to zero as n tends to infinity, we see that our assumptions hold with high probability if we choose r as in (6) for any $\delta > 0$. In other words, we need to choose $r^2 = cn$ for a suitable constant $c > 0$. With this choice of r, it follows that routing table sizes are quadratic in n, the logarithm of the number of nodes, and that the complexity of the join and leave protocols are also quadratic in n.

Finally, we remark on the choice of transmission range. It was shown by Gupta and Kumar [8] that the transmission range of a node, defined as $2r$, must satisfy $4\pi r^2 \geq n \log 4$ in order for $G(4^n, 2r)$ to be connected. Clearly, this is a minimum requirement for routing! Thus, our routing protocol only requires a transmission range which is some constant multiple of the minimum range needed for connectivity; it has the same scaling behaviour in relation to the number of nodes.

We also remark that while $r^2 = cn$ is the natural scaling relationship, there is room to choose the constant c in order to trade off between stretch and storage. From the analysis above, the average routing table size is $4\pi r^2(n+1)$, so larger values of r result in larger routing tables. On the other hand, each iteration of Algorithm A is likely to match more digits of the destination prefix, leading to a path with fewer intermediate nodes, and hence with smaller stretch, on average. These observations are confirmed by simulation results presented in the next Section.

III. Simulation Results

In this section, we carry out NS-2 [18] simulations to evaluate the efficiency of our scheme and present the results.

In the simulations, 400 fixed nodes are randomly uniformly distributed over a 2000x2000 square meters area. We use the CMU wireless extension to simulate wireless ad-hoc nodes and their communications. Every node knows its geometric location, has a radio range of 250 meters, runs our routing algorithm on it, and randomly selects an ID in $\{0, 1, 2, \ldots, 399\}$.

In the simulations, we randomly select 700 pairs of source and destination nodes to compute stretch. For the results reported in this section, we define stretch as the ratio between the number of hops along the path selected by our routing algorithm, and the one along the shortest path. As remarked earlier, we use simulations to evaluate the stretch of our scheme with respect to graph distance, i.e. hop count; this complements the analytical results, which pertain to Euclidean distance. The sizes of nodes' routing tables are also of interest. Finally, we evaluate the impact of different choices of r on the stretch and routing table size, which are our two main performance measures.

The average stretch and size of routing table for $r = 250, 310, 510$ and 810 meters are shown in Fig. 1. For $r = 250$ meters, which is the radio range of a node and hence a natural choice, our scheme achieves an average stretch around 1.45 and average number of routing entries per node around 40 (10% of the total number of nodes). As r increases, the average size of routing tables increases, while at the same time the average stretch decreases. For $r = 510$ meters, the average stretch is 1.27 and the average number of routing entries is 109. This demonstrates the trade-off between stretch and size of routing table, and confirms the observations made in Section II-E on the effect of different values of r.

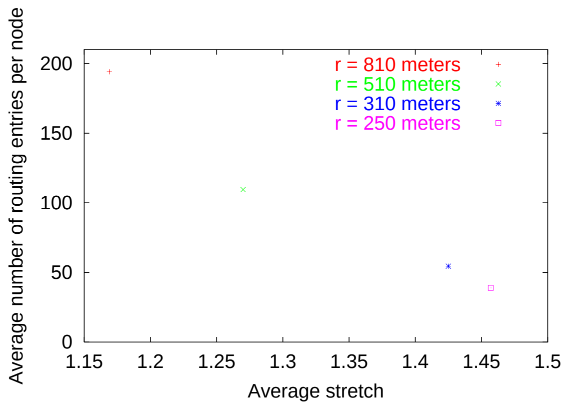

Fig. 1. Trade-off between stretch and size of routing table per node.

To provide a more detailed view of the performance metrics, we show cumulative probability statistics of stretch in Fig. 2, and of routing table sizes in Fig. 3. For example, in the case where $r = 250$ meters, about 99% of the sampled routes have a stretch less than 2.7, and 99% of nodes have fewer than 60 routing table entries. We also observe that the variance of routing table sizes increases as r increases. This is because edge effects when constructing Level k routing entries in $B(\cdot, 2^k r)$ are more significant for larger values of r, and so the difference in the number of routing entries between nodes close to the edges and those near the centre becomes larger.

To evaluate the join and leave operations, we fix $r = 510$ meters, start our routing scheme on every node, and wait until the network stabilizes. Then we randomly pick 10 nodes to leave, and rejoin the network at the same geometric positions. (We do not pick a large number of nodes to perform leave and join operations simultaneously. This is because the network isolates into several partitions instead of remaining connected,

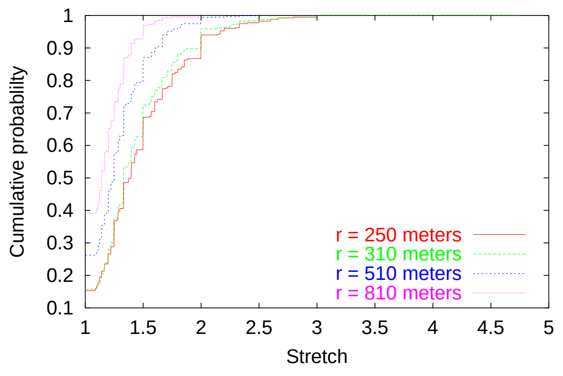

Fig. 2. Cumulative probability statistics of stretch.

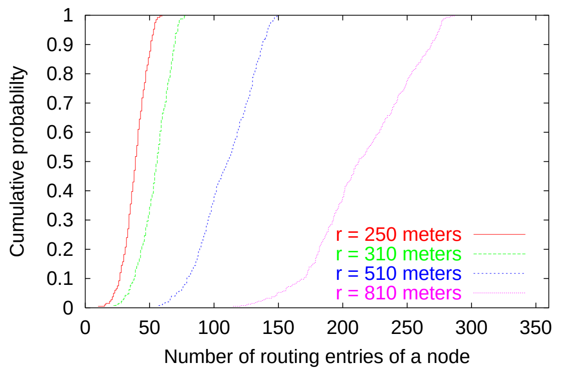

Fig. 3. Cumulative probability statistics of nodes' number of routing entries.

if too many nodes leave at the same time.) The routing tables of all nodes before the leave and after the join are compared, and we find all contents are matched. This indicates the correctness of the join and leave operations described in Section II-D. Furthermore, the average number of unicast messages sent by those 10 nodes is 132, which is close to 109 (average size of routing table) + $12 \cdot \log 400$ (retrieving routing entries) as calculated using results in Section II-D.

IV. Discussion and Conclusions

We presented a compact routing scheme for wireless ad-hoc networks. The scenario considered is one where nodes are aware of their own geographic location, but not of that of other nodes. We proposed a scheme which provides a distributed name (node identifier) to location directory. It requires polylogarithmic (in the total number of nodes) storage at each node, and enables routing between arbitrary source and destination identifiers with constant stretch. Join and leave operations are lightweight, incurring only polylogarithmic cost in terms of the number and size of messages required. The results are obtained for a model of nodes located uniformly at random on a square, and hold with high probability in this model.

Our scheme is somewhat similar to that of Abraham and Malkhi [2], in that it uses the same basic idea of prefix based name-location lookup. By imposing much stronger model assumptions, our scheme achieves a better stretch-storage trade-off; more importantly, it permits light-weight join and leave operations. We believe that the assumptions are not altogether unrealistic for the target application scenario, and that the comparative ease of dealing with changes to the network is important in this setting.

Future work could aim to extend the scheme to support mobility, as well as extending it to perform multi-path routing. It would also be of interest to obtain similar results with weaker assumptions on the distribution of nodes.

References

[1] I. Abraham, C. Gavoille, D. Malkhi, N. Nisan and M. Thorup, "Compact name-independent routing with minimum stretch", in *Proc.* 16th *ACM Symp. Parallel Algo. Arch. (SPAA)*, 2004.

[2] I. Abraham and D. Malkhi, "Name Independent Routing for Growth Bounded Networks", in *Proc.* 17th *ACM Symp. Parallel Algo. Arch. (SPAA)*, 2005.

[3] I. Abraham, C. Gavoille, A. Goldberg and D. Malkhi, "Routing in Networks with Low Doubling Dimension", *Microsoft Research Technical Report No. MSR-TR-2005-175*, Dec. 2005.

[4] M. Arias, L. J. Cowen, K. A. Laing, R. Rajaraman and O. Taka, "Compact routing with name independence", in *Proc.* 15th *ACM Symp. Parallel Algo. Arch. (SPAA)*, 2003.

[5] A. Busson, G. Chelius and E. Fleury, "From Euclidean to hop distance in multi-hop radio networks: a discrete approach", *INRIA Research Report RR 5505*, 2005.

[6] P. Fraigniaud and C. Gavoille, "Routing in trees", in *Proc.* 28th *Intl. Colloq. on Automata, Languages and Programming (ICALP)*, 2001.

[7] C. Gavoille and D. Peleg, "Compact and localized distributed data structures", *J. Distrib. Comp.* 16: 111–120, PODC 20-year Special Issue, 2003.

[8] P. Gupta and P. R. Kumar, "Critical Power for Asymptotic Connectivity in Wireless Networks", pp. 547-566, in *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W.H. Fleming.*, Edited by W.M. McEneany, G. Yin, and Q. Zhang, Birkhauser, Boston, 1998. ISBN 0-8176-4078-9

[9] P. Jacquet, P. Mühlethaler, T. Clausen, A. Laouiti, A. Qayyum, L. Viennot, "Optimized link state routing protocol for ad hoc networks", in *Proc. IEEE INMIC*, 2001.

[10] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks", in *Mobile Computing*, T. Imielinksi and H. Korth eds., Kluwer Academic Publishers, 1996.

[11] B. Karp and H. T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks", in *Proc. Mobile Computing and Networking (Mobicom)*, 2000.

[12] T. Moscibroda, R. O'Dell, M, Wattenhofer and R. Wattenhofer, "Virtual Coordinates for Ad hoc and Sensor Networks", in *Proc.* 2nd *Joint Workshop on Foundations of Mobile Computing (DIALM-POMC)*, 2004.

[13] D. Peleg and E. Upfal, "A trade-off between space and efficiency for routing tables", *Journal of the ACM* 36: 510–530, 1989.

[14] C. E. Perkins, E. M. Belding-Royer, and S. Das, "Ad Hoc On Demand Distance Vector (AODV) Routing", *IETF RFC 3561*, 2003.

[15] C. G. Plaxton, R. Rajaraman and A. W. Richa, "Accessing Nearby Copies of Replicated Objects in a Distributed Environment", *Proc.* 9th *ACM Symp. Parallel Algo. Arch. (SPAA)*, 1997.

[16] M. Thorup and U. Zwick, "Compact routing schemes", in *Proc.* 13th *ACM Symp. Parallel Algo. Arch. (SPAA)*, 2001.

[17] F. Xue and P. R. Kumar, "On the θ-coverage and connectivity of large random networks". Preprint, 2005.

[18] Network Simulation version 2, *http://www.isi.edu/nsnam/ns/*