Wireless Networks 11, 607–618, 2005

Flooding Strategy for Target Discovery in Wireless Networks

ZHAO CHENG and WENDI B. HEINZELMAN
Department of Electrical and Computer Engineering, University of Rochester, Rochester, NY 14627, USA

Abstract. In this paper, we address a fundamental problem concerning the best flooding strategy to minimize cost and latency for target discovery in wireless networks. Should we flood the network only once to search for the target, or should we apply a so-called "expansion ring" mechanism to reduce the cost? If the "expansion ring" mechanism is better in terms of the average cost, how many rings should there be and what should be the *radius* of each ring? We separate wireless networks based on network scale and explore these questions. We prove that two-ring and three-ring schemes can reduce the cost of flooding compared to a single attempt, and we provide a general formula to determine good parameters for the two-ring and three-ring hop-based flooding schemes. Through simulations, we show that choosing flooding parameters according to our techniques gives performance close to that of ideal flooding schemes. Afterwards, we extend our work from the single target discovery problem to the multi-target discovery problem. We show that a properly chosen searching radius can save much more searching cost than a simple radius selection scheme for multi-target discovery problems.

Keywords: single/multi target discovery, mathematical induction, expansion ring

Abbreviations: EXP—EXPansion ring scheme; DSR—Dynamic Source Routing; AODV—Ad hoc On-Demand Distance Vector routing

1. Introduction

Flooding is a basic operation and has extensive applications in target discovery in wireless networks, such as those widely utilized in the route discovery process in several routing protocols [8,12], those used in wireless sensor networks for sensor discovery [7], or those used in wireless ad hoc networks for service discovery [16]. Query packets are flooded inside the network to search for a certain target node. When the target node receives the query packet, it responds to the source node, not only to inform the source node about its existence, but also to avoid further unnecessary flooding attempts from the source node.

However, even if the flooded packet reaches the target, packets flooded towards other directions continue. Different types of networks introduce different methods to control the flooding. Flooding may be controlled by setting the hop limit, which guarantees a packet will not be transmitted more than the maximum number of hops, or by setting a distance limit from the source, which guarantees a packet will not be transmitted beyond a certain geographical limit.

Using hop limit, the authors of DSR consider a mechanism called "expansion ring" to search for a target [8] (see Section 2 for a description of expansion ring). The authors claim that in this way, a node can explore for the target progressively without flooding the entire network, and the only drawback of this scheme is the increasing latency due to multiple discovery attempts. However, DSR applies a simpler scheme that searches the one-hop neighbors first and then the entire network. The idea of the expansion ring was implemented later in AODV [12]. An interesting question is whether or not using the expansion ring technique always reduces flooding overhead compared with flooding the network just once. If not, when and how should the expansion ring technique be applied?

We can generalize this question as follows. With a certain restriction on the searching scheme, either the maximum hop limit or the longest distance from the source node, how many attempts should we take to achieve minimum cost or latency to find the target? If multiple attempts are better than one, what should be the *radius* for each of these attempts? What if we are looking at small-scale networks in which a node has no restrictions in flooding and is allowed to search the entire network? More generally, what is the optimal searching scheme if at least k targets are required to be discovered from multiple identical targets?

In this paper, we explore these questions and provide analytic solutions for each question. We mainly focus on the first question, which is to find only one target with the smallest cost. We briefly investigate the other questions, but due to space constraints, we limit our discussion of these topics. Our contributions are more than just solving these problems. First, we propose a general framework to model and analyze flooding schemes in wireless networks. Second, most of our conclusions and algorithms can be directly applied to existing networks and protocols. Finally, we clarify that the "expansion ring" scheme can reduce the searching cost only under certain conditions, and they are usually non-optimal. To the best of our knowledge, this is the first formal study undertaken to compare different searching strategies.

The rest of this paper is organized as follows. Section 2 provides an overview on the previous efforts in reducing discovery overhead and some other related work. Section 3 presents the analytic framework and procedures we will use throughout this paper. In Section 4, we address the flooding problem in large-scale networks. In Section 5, we extend the problem to small-scale networks with no restrictions, that is, nodes may

flood the entire network if necessary. Section 6 provides extensive simulations and compares the performance of our schemes with existing schemes and ideal schemes in terms of both cost and latency. Section 7 extends the problem to a multi-target discovery problem. Section 8 concludes the paper.

2. Related work

The most common use of target discovery is in routing protocol implementations. Typical examples are DSR [8] and AODV [12]. In both protocols, nodes search for a target gradually in order to avoid flooding the entire network. The procedure in DSR is relatively simple. A node searches its one-hop neighbors first, and if the target is not found, the node then searches the entire network. AODV uses a different approach, where a node increases its searching radius linearly from an initial value until it reaches a predefined threshold. After that, a network-wide search has to be performed. Both of these expansion ring schemes assume that there are route caches residing in the nodes. When the route caching condition is weak and node mobility is high, these schemes will not reduce the search overhead compared with flooding the entire network once as effectively as expected. Instead, an improper utilization of the expansion ring scheme will lead to even more overhead. How to discover a single target efficiently is the main topic of this paper.

There has also been some work on target discovery in sensor networks, such as ACQUIRE [14] and rumor routing schemes [1]. ACQUIRE avoids flooding and traditional query/response stages by refining the query into sub-queries and resolving each sub-query by means of local searching and random forwarding. Rumor routing manages to find an optimal balance point between query flooding and event notifications flooding. Both these two algorithms show a better performance than the basic flooding search algorithms. However, both of them are confined to certain circumstances and are application-specific. In contrast, the conclusions presented in this paper can be applied to general target discovery scenarios.

There are some other routing protocols that reduce discovery overhead by assuming there are some location-based devices to aid the routing [18]. The cost and the inaccuracy of these devices confine the implementation of these protocols. In our paper, we do not assume the existence of such devices.

Multi-target discovery, which is to find at least k targets from m members, also has extensive applications in wireless networks. Examples that require a mandatory multi-target discovery are NTP (network time protocol) [10], ITTC (Intrusion Tolerance via Threshold Cryptography) [17], sensor localization [2], and sensor information collecting [11]. Examples that require a multi-target discovery for robustness are NIS (network information system) [15], NTP [10] and any application requiring auxiliary backups. Examples that require a multi-target discovery for load distribution are peer-to-peer systems [4] and distributed computing systems [13].

3. Assumptions, terminology and analysis methodology

Before Section 7, we focus on the single target discovery problem, which is to find only one target from the area of interest. Multi-target discovery will be addressed in Section 7.

All of the analysis in Sections 4 and 5 is based on blind flooding. In this basic flooding scheme, nodes forward a packet once and only once, only if they are not the destination of the packet and the nodes are allowed to forward, e.g., the hop limit is not zero. Also, we ignore the potential increase of the packet length and the cost it brings during packet propagation. For simplicity, we also ignore potential packet collisions, which can be effectively decreased by inserting a random delay time before forwarding. Also, we assume that the first flooded packet that arrives at a certain node follows the shortest path. This assumption is valid since packets following the shortest path usually need less transmission time and delay.

During our analysis, we assume we are studying a snapshot of the network and nodes are static during the analysis. However, even if nodes are mobile, there are several reasons that our analysis is still valid. First, the flooding searching time is short and nodes will not move too far away. Second, we are looking at broadcasting which does not have the problems of unicasting such as link failures. Third, since nodes are moving randomly and independently, the number of nodes in a certain region is stable and will not have adverse effects on our analysis.

The main part of this paper is organized according to the following methodology. We classify wireless networks into two categories, large-scale networks and small-scale networks, which provide different assumptions for our analysis. For large-scale networks, we ignore edge effects and assume every node is identical; while for small-scale networks, nodes are no longer the absolute *center* of the flooded area and edge effects have a significant impact on the desired flooding scheme.

We study hop-based flooding control, where the source node will set a hop limit that a flooded packet can reach. Upon receiving this packet, intermediate nodes decrement the value and check the value. If the value becomes zero, the node will discard the packet. If not, the node will forward the packet with the decremented value.

We will focus on two metrics: cost and latency. Cost is defined as the total number of packets transmitted and is closely related to the consumed energy. Latency is defined as the total time that the source node takes to receive a response from the target node.

Besides theoretical interests, most of our analysis results can be directly employed into realistic wireless network applications and protocols. For example, the analysis results for hop-based flooding in large-scale networks can be employed in DSR route discovery and can even be extended to wired networks, and the results for hop-based flooding in small-scale networks can be applied to service discovery protocols in ad hoc networks.

For quick reference, we list our terms and notations in this paragraph. *Once-for-all* is a strategy that nodes only flood once to discover a target. *n-ring* is a strategy that nodes attempt at

Table 1
Notations used throughout this paper.

C^n	Cost of an n-ring scheme
L^n	Latency of an n-ring scheme
h_i	Number of hops
M	Maximum hop number
N_T	Total number of nodes
N_i	Number of nodes exactly i hops away
$D_{i,j}$	Cost difference between an i-ring and a j-ring scheme
ρ	Node density
A	Area of a flooded region
R_t	Node transmission range

most n times to discover a target. *Two-ring* and *three-ring* are special n-ring strategies and are fully studied. Other notations are listed in Table 1.

4. Single target discovery: Restricted search in large-scale networks

In this section, we consider the flooding strategy problem in large-scale networks. The most important property of this type of network is that flooding is limited to a small area compared to the whole network. The border of the network is far away from the reaching range of the flooding process, so edge effects are ignored during analysis.

In a general case, nodes adopt a hop limit scheme to control the flooding. Let us model the question under this hop-based assumption as shown in figure 1. Please refer to [3] for details with geography-based flooding.

A large number of nodes are placed uniformly and independently in a two-dimensional space $\mathbb{R}^2$ with node density ρ. A node wants to search nodes within the hop limit M. Suppose the number of nodes that are exactly i hops away from the source node is N_i, for $i \in \{1 \dots M\}$.

Suppose we apply an n-ring scheme, $n \geq 1$. For the ith attempt, we set the hop limit to h_i. Thus we have a corresponding hop set $H = \{h_i, i \in \{1 \dots n\}, h_i \in \{1 \dots M\}\}$. The source node will start searching by setting the hop limit to h_1; if it fails for the ith attempt, it will set the searching limit to h_{i+1} and take the $(i+1)$st attempt, until searching for the last time by setting the hop limit to $h_n = M$.[1] The question becomes: what is the optimal number of attempts n, and what is the optimal set H to achieve minimum average cost and minimum average latency, given a specific n?

To aid our analysis, we define $N_0 = 1$ and $h_0 = 0$, which can be understood as the number of nodes that are 0 hops away from the source node is 1, the source node itself.

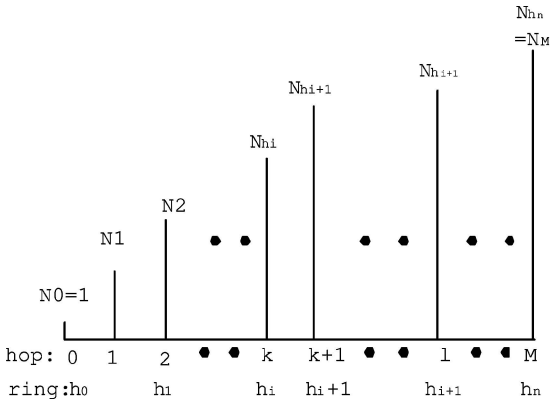


Figure 1. Model of target discovery for large-scale wireless networks using hop-based flooding control.

4.1. Cost

To flood with hop count set to h_k, the average cost equals the number of nodes whose distance is less than h_k hops. Note that nodes that are exactly h_k hops away will not forward the packet and are not taken into account for cost calculation. That is, $co_k = \sum_{j=1}^{h_k} N_{j-1}$.

Suppose the target node is h_d hops from the source node and $h_i < h_d \leq h_{i+1}$, for $i \in \{0 \dots n-1\}$. To find the target, the source node has to fail for the first i attempts and succeed at the $(i+1)$st attempt. Thus the total cost is $C_i = \sum_{k=1}^{i+1} co_k = \sum_{k=1}^{i+1} \sum_{j=1}^{h_k} N_{j-1}$.

If the target is in the searching area and the number of nodes inside the ring i and $i+1$ is $\sum_{m=h_i+1}^{h_{i+1}} N_m$, the probability that the target node hop h_d is within h_i and h_{i+1} is:

$$\mathcal{P}_i = \mathcal{P}\{h_i < h_d \leq h_{i+1}\} = \frac{\sum_{m=h_i+1}^{h_{i+1}} N_m}{\sum_{m=1}^{M} N_m}.$$

The average cost C^n to search for a random node using an n-ring scheme is:

$$C^n = \sum_{i=0}^{n-1} \mathcal{P}_i C_i = \sum_{i=0}^{n-1} \left[\frac{\sum_{m=h_i+1}^{h_{i+1}} N_m}{\sum_{m=1}^{M} N_m} \sum_{k=1}^{i+1} \sum_{j=1}^{h_k} N_{j-1} \right] \quad (1)$$

N_T is the total number of nodes inside the M-hop searching area, including the source node, and equals $\sum_{m=0}^{M} N_m$.

We start by studying the cost when $n = 1$.

1. $n = 1$. In this once-for-all scheme, all the nodes transmit except those that are exactly M hops away. Thus, we have the cost $C^1 = N_T - N_M$.
2. $n = 2$. Suppose we set hop limit $h_1 = k$ for the first attempt, and for the second attempt we set hop limit $h_2 = M$. To simplify the equations, let us define $a_k = \sum_{i=1}^{k} N_i$, which is the total number of nodes within k hops of the source, $k \geq 1$. From Equation (1), we have

$$\begin{aligned} C^2 &= \mathcal{P}_1 C_1 + \mathcal{P}_2 C_2 \\ &= \frac{a_k}{N_T - 1}(N_0 + a_k - N_k) \\ &\quad + \frac{N_T - 1 - a_k}{N_T - 1}[(N_0 + a_k - N_k) + (N_T - N_M)] \end{aligned}$$

[1] Note the difference between h_{i+1} and $h_i + 1$, as they may look similar in the text.

$$= \frac{1}{N_T - 1}(2(N_T - 1) + (N_T - 1)^2 - N_M(N_T - 1) - (N_T - 1)N_k + a_k N_M - a_k) \quad (2)$$

Subtracting C^1 from C^2, we have the difference between the two-ring scheme and the once-for-all scheme

$$D_{2,1} = C^2 - C^1 = \frac{1}{N_T - 1}(N_T - 1 - (N_T - 1)N_k + a_k N_M - a_k) \quad (3)$$

If $D_{2,1} < 0$, which means $C^2 < C^1$, the two-ring scheme is preferred; otherwise, the once-for-all scheme is better. The only variable of $D_{2,1}$ is k, which is the hop number for the first searching attempt, and all the other parameters such as N_M and N_T are constants for a given network. Now we want to determine if there is some k that enables $D_{2,1} < 0$, and if so, what the optimal k should be to achieve min $D_{2,1}$.

After placing a large number of nodes in a disk of unit radius and determining the number of the nodes within the first k hops from the center node, we found that before edge effects occur, the number of nodes at a certain hop distance from the source is roughly linear with hop number.[2]

Thus, we can estimate $N_i \approx Bi$ for large scale networks, where B is a constant value larger than 1 and is closely related to the network density ρ. Thus we estimate the sequence $N_0, N_1, \ldots, N_M$ as $1, B, 2B, \ldots, MB$. Using this estimation, the total number of nodes inside M hops, N_T, equals $\sum_{i=0}^{M}(Bi)$, and the cost difference equation (3) finally becomes $D_{2,1} = \frac{B(k-M)(-1-M+k(-1+BM))}{M(M+1)}$.

$D_{2,1}$ is a parabola function with k ranging from 1 to $M - 1$. By analyzing this function, we reach two conclusions. First, when $k_{opt} = \lceil \frac{M}{2} \rceil$,[3] $D_{2,1}$ achieves its minimum value, and this value is less than zero. Thus, by applying a two-ring scheme with k_{opt} as the first attempt hop number, we can achieve less cost than the once-for-all scheme. Second, when $k = 1$, $D_{2,1}$ reaches its maximum $\max D_{2,1} = \frac{B(-1+M)(2+M-BM)}{M(M+1)}$. This value is less than zero when $B > 1$.

Based on our estimation, we conclude that the costs of all the two-ring schemes are less than the cost of the once-for-all scheme. Furthermore, the cost of a two-ring scheme reaches its minimum when the first attempt hop limit is set to $\lceil \frac{M}{2} \rceil$. The cost reaches its maximum among all the two-ring schemes when the first attempt hop limit is set to 1.

3. $n = 3$. In a three-ring scheme, there are two parameters to adjust, the first attempt hop limit h_1 and the second attempt hop limit h_2. Let us look at a special scheme in which $h_1 = 1$ first. Using similar procedures as above, we find that the best performance occurs when we choose $h_2 = \lceil \frac{M+1}{2} \rceil$. Also, we find that the cost of this $(1, \lceil \frac{M+1}{2} \rceil, M)$ scheme is even lower than the cost of an $(\lceil \frac{M}{2} \rceil, M)$ scheme when $B > 4$.

   In a more general case, h_1 does not equal 1. However, we cannot prove that the special three-ring scheme of $h_1 = 1$ is the best among all the three-ring schemes.

4. $n \geq 2$. Also, we are not able to prove that an $(n + 1)$-ring scheme derived from an n-ring scheme is always worse. From the final equation of the cost difference equation between these two schemes, we notice that there is one negative term on the order of h_i^3, while there are two positive items on the order of h_j^4 for $1 \leq h_i, h_j \leq M$. This indicates that there are very few choices for a derived $(n + 1)$-ring scheme to be better than an n-ring scheme in terms of cost.

In this part, we have studied hop-based flooding schemes in large scale networks in terms of cost. Let us summarize our conclusions.

1. To obtain a good two-ring scheme, the first hop limit should be set to $\lceil \frac{M}{2} \rceil$ and the second hop limit should be set to M. We have proven that this two-ring scheme has less cost than the once-for-all scheme and is optimal for all the two-ring schemes.
2. To obtain a good three-ring scheme, set the first hop limit to 1, the second hop limit to $\lceil \frac{M+1}{2} \rceil$ and the third hop limit to M. We have proven that this three-ring scheme has even less cost than the optimal two-ring scheme.
3. However, we cannot prove that our three-ring scheme is optimal among all the three-ring schemes. We also cannot prove that it leads to higher cost by splitting an n-ring scheme to an $(n+1)$-ring scheme. However, we conjecture that it is quite probable that any $(n + 1)$-ring scheme will have a larger cost than the n-ring scheme from which it was derived for $n \geq 3$.
4. The scheme that is applied currently in DSR, which is to set the hop limit to 1 for the first attempt and M for the second attempt, can be seen as one of the two-ring schemes. Based on our results, we show that this approach leads to the highest cost among all the two-ring schemes.

For geography-based flooding, using similar analytical methods as above, we can easily prove that the once-for-all scheme has the same cost as the two-ring scheme, and this cost is the lowest among all the costs of possible n-ring schemes. These conclusions are valid for both large-scale and small-scale networks [3].

4.2. Latency

Let us define a constant T as the time interval from when a node receives a packet to when it finishes forwarding the packet. Then the time a flooded packet takes to reach a node l hops away is lT. Suppose the target node is i hops from the source node and $h_k < i \leq h_{k+1}$, for $k \in \{0 \ldots n - 1\}$. To find the target, the source node has to fail for the first k times, which takes $2h_m T$ time for the mth attempt $m \in \{1, \cdots k\}$, and

[2] This can be seen from part (3) and (4) of figure 3 when the source node is located at the network center with the distance to the network border $x = 1$. The node distribution shows a linear tendency with small numbers of hops before edge effects occur.

[3] $\lceil x \rceil$ means the smallest integer greater than or equal to x.

succeed at the $(k+1)$st attempt, which takes $2iT$ time. Note that each attempt time is doubled because the source node has to wait enough time for a potential acknowledgement from the target. Thus we have the total latency to search for a target i-hops away is $L_i = \sum_{m=1}^{k} 2h_m T + 2iT$.

Since N_i is the total number of nodes exactly i hops away from the source, the probability that a node is i hops away from the source node is $\mathcal{P}_i = \frac{N_i}{\sum_{i=1}^{M} N_i} = \frac{N_i}{N_T - 1}$, for $i \in \{1 \ldots M\}$. The average latency L to search for a random node is:

$$L = \sum_{i=1}^{M} \mathcal{P}_i L_i = \left[\sum_{i=1}^{M} 2N_i i + \sum_{k=1}^{n-1} \sum_{i=h_k+1}^{h_n} (2h_k N_i) \right] \frac{T}{N_T - 1} \quad (4)$$

Applying mathematical induction on equation (4), we can easily prove that for all the $(n+1)$-ring schemes with $n \geq 1$, we can find an n-ring scheme that has a shorter latency. Furthermore, we can prove that the once-for-all scheme has the shortest latency of all the schemes (proof omitted).

5. Single target discovery: Unrestricted search in small-scale networks

In the previous section, we assumed that during the discovery process, nodes are unwilling to flood the whole network and have certain restrictions on the maximum *region* to be covered. In this section, we extend the model to small-scale networks in which nodes may search the whole network for a target. The main difference between this model and the previous model is that nodes are no longer the absolute center of the flooded region and edge effects must be taken into account during analysis. In other words, the node distribution at certain hops away from the source node no longer shows a linear tendency and is closely related to the source node's position.

In this section, we will confine our discussion to the once-for-all and the two-ring schemes. The average cost is slightly different from that in Section 4. Not knowing what the largest hop number is, in order to guarantee that the whole network is covered, the source node has to apply a large enough hop limit number for the last attempt. The direct effect is that nodes at the maximum hops also have to forward the packet; while in Section 4, nodes at M hops do not forward the packet.

As before, we look at the cost from $n = 1$.

1. $n = 1$. All the nodes have to forward and the cost $C^1 = N_T$ (note the difference with that in large-scale networks).
2. $n = 2$. The cost of a two-ring scheme with k as the first attempt hop number is

$$C^2 = \frac{1}{N_T - 1}[a_k(1 + a_k - N_k) + (N_T - a_k - 1)(1 + a_k - N_k + N_T)] \quad (5)$$

where $a_k = \sum_{i=1}^{k} N_i$, which is the total number of the first k hops nodes, as defined earlier.

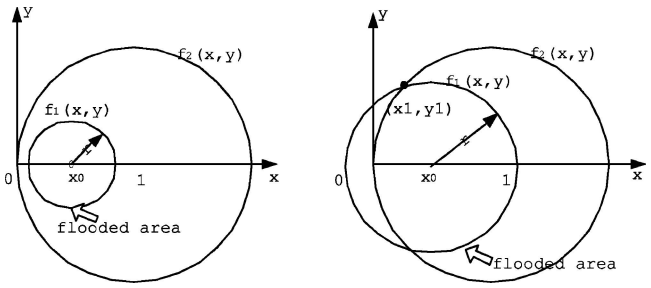


Figure 2. The intersection area is the region covered by flooding. When $r < x_0$, it is $f_1(x, y)$; when $r \geq x_0$, it is the intersection of two areas.

Whether to use the once-for-all scheme or a two-ring scheme depends on $D_{2,1}$, the cost difference between these two schemes.

$$D_{2,1}(k) = C^2 - C^1 = [(N_T - 1) - a_k - N_k(N_T - 1)]\frac{1}{N_T - 1} \quad (6)$$

If we apply $k+1$ for the first attempt hop limit instead of k, the difference becomes

$$D_{2,1}(k+1) = \frac{(N_T - 1) - a_k - N_{k+1} - N_{k+1}(N_T - 1)}{N_T - 1} \quad (7)$$

If $D_{2,1}(k+1) < D_{2,1}(k)$, a two-ring scheme applying $k+1$ is preferred over a two-ring scheme applying k. Subtracting equations (7) and (6), we have

$$D_{2,1}(k+1) - D_{2,1}(k) = [N_k(N_T - 1) - N_{k+1}N_T]\frac{1}{N_T - 1} \approx N_k - N_{k+1} \quad (8)$$

As can be seen, as long as $N_k < N_{k+1}$, we should apply a two-ring scheme using $k+1$ as the first attempt hop limit instead of k. This trend continues until N_k starts to becomes larger than N_{k+1}. To determine this turning point, an estimation of the sequence N_i is necessary.

Just as we estimated N_i in Section 4, we provide a general algorithm for the sequence estimation of nodes at different locations. We set up two-dimensional coordinates as shown in figure 2. The node is located at x_0 away from the network border. When the flooding radius is large, part of the potential flooding area exceeds the edge of the network. In figure 3, we show the geographic overlapping area $A(r)$ and the number of nodes at different hop numbers when the source node is located at different distances x_0 from the network border. We notice that the node distribution in the lower part is just like the sampling of the derivative of the continuous geographic overlapping area shown in the upper part. Thus, we only need to estimate the maximum hop number and do proper samplings at each hop number. Due to space constraints, we omit the details here. We suppose that given the node's distance from the border x_0 and the network parameters of the total

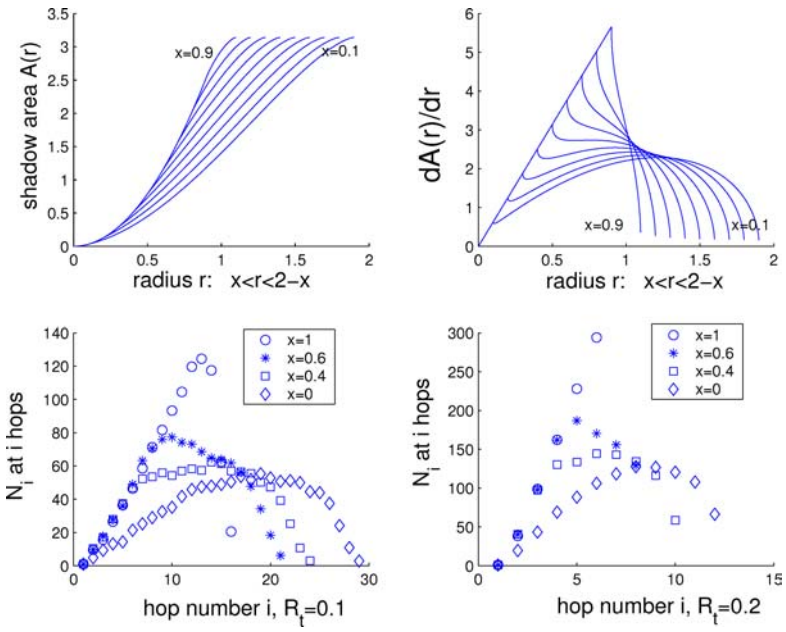


Figure 3. Area $A(r)$ of flooded region and the relation between the number of nodes N_i exactly i hops away with $\frac{dA(r)}{dr}$. Part (1) is the flooded area $A(r)$; part (2) is $\frac{dA(r)}{dr}$; part (3) is the N_i in a network of 1000 nodes and transmission range 0.1; part (4) is the N_i in a network of 2000 nodes and transmission range 0.2. Part (3) and part (4) look like a sampling of part (2).

number of nodes N_T and the node transmission range R_t, we have the estimated sequence $\tilde{N}_{i,x_0}$ of N_i for a node at location x_0.

5.1. Self-location aware

First, suppose a node knows its distance to the border of the network x_0 and can adjust its own hop limit k for the first attempt. Here is our proposed two-ring scheme for this case: the node first estimates $\tilde{N}_{i,x_0}$ based on its location x_0. Then it finds out from the estimated sequence $\tilde{N}_{i,x_0}$ the value of i where $\tilde{N}_{i,x_0} \approx \tilde{N}_{i+1,x_0}$ and sets k to this value for the first attempt. If this fails, the node must pick a large enough number for the second attempt to ensure that the flooded packet reaches all nodes in the network.

5.2. Self-location unaware

More realistically, nodes do not have knowledge of their location in the network. Every node has to apply the same flooding strategy and set the same predetermined values for each attempt. Now we consider how to minimize the cost from the system view.

First, we can prove that for a uniformly distributed network, the Probability Distribution Function (pdf) of a random node located X away from the border is

$$f_X(x) = 2(1-x) \quad \text{for } 0 < x \leq 1 \tag{9}$$

For a node located at x, after the estimation of its node number sequence, we can correspondingly calculate the cost $C(x,k)$ of a two-ring scheme with first attempt k from equation (5). Since a consistent strategy has to be applied, suppose every node sets k_{sys} as its first attempt. We find the overall cost for the whole system $C_{sys}(k_{sys})$ as

$$\begin{aligned} C_{sys}(k_{sys}) &= \int_0^1 f_X(x) C(x, k_{sys})\, dx \\ &= \int_0^1 2(1-x) C(x, k_{sys})\, dx \end{aligned} \tag{10}$$

Based on equations (5) and (10), we propose our two-ring scheme here. First, gather enough samples of x and estimate the sequences $\tilde{N}_{i,x}$ for each sample of x. Then from equation (5), calculate the two-ring cost sequence $C^2(x,k)$ of each sample x. Put each $C^2(x,k)$ into equation (10) and calculate the system cost $C_{sys}(k)$. Finally, determine the global point k_{opt} as the point where the minimum C_{sys} is achieved. The calculation of this good k_{sys} can be implemented during the network design phase and input to each node as a system parameter—nodes do not need to calculate this value on the fly.

Overall, we propose a good two-ring scheme to reduce the cost for hop-based small-scale networks. For the first attempt, if a node has knowledge of its current position with regard to the network boundary, it can set its first hop limit from the estimation of $\tilde{N}_{i,x}$ to minimize cost. If it has no such information, it should set the number to the pre-calculated good value to minimize overall system cost. For the second attempt, nodes just need to set a large enough hop limit to cover the entire network.

6. Simulations

6.1. Goals, metrics and simulation models

We have drawn several conclusions in Sections 4 and 5. The first goal of our simulations is to verify our conclusions and conjectures. Since we have proposed some good schemes based on the analysis of estimated N_i, another goal of our simulations is to verify the accuracy of the estimation and the performance of these schemes by determining how far away our proposed schemes are from the ideal schemes. The ideal n-ring scheme is found by thoroughly testing all the possible n-ring schemes through simulations on randomly generated scenarios.

In each simulation section, we compare the cost savings per target search and the latency performance of different schemes. We measured the cost savings of each scheme compared to the basic once-for-all scheme whose cost is constant for a fixed scenario. This metric indicates how much improvement we can achieve by replacing the once-for-all scheme with the investigated scheme. Every test is repeated on 50 different random scenarios, and the results are averaged.

The schemes that are tested are: the once-for-all scheme, the expansion ring scheme, the DSR scheme, our two-ring scheme, our three-ring scheme, the ideal two-ring scheme, the ideal three-ring scheme and the ideal four-ring scheme. The expansion ring scheme is an n-ring scheme, which sets the first attempt hop limit to 1 and doubles the hop limit upon each failure until the maximum restriction M is reached. The current DSR scheme can be seen as a two-ring scheme with the first attempt hop limit set to 1. The choice of our schemes varies for different types of networks and will be specified in each individual simulation part.

6.2. Performance comparison for large-scale networks

In this section, we compare the cost and latency performance of different schemes in hop-based large-scale networks. For these simulations, 20000 or 40000 nodes with transmission range $R_t = 0.03$ are placed in a unit radius disk. The center node searches for a random target from nodes within M hops away. We test for $M = 12$ and $M = 16$; with these values of M, the flooding area is far away from the border of the whole large networks and no edge effects need to be taken into account.

Our proposed two-ring scheme is to set the first hop limit to $\lceil \frac{M}{2} \rceil$ and the second hop limit to M. Our proposed three-ring scheme is to set the first hop limit to 1, the second hop limit to $\lceil \frac{M+1}{2} \rceil$ and the third hop limit to M. We also measure the performance of all possible two-ring, three-ring and four-ring schemes and pick the minimum of each scheme as the ideal value for these n-ring schemes.

Figure 4 shows the results. As can be seen from this figure, the expansion ring's savings are less than zero, that is, it costs even more than the basic once-for-all scheme. The DSR scheme, as a member of the two-ring scheme family, does have some savings over the once-for-all scheme. However, the savings are low. The reason is that it is the worst of all the two-ring schemes, as proven in Section 4. Our two-ring scheme has less cost saving than our three-ring scheme. However, their difference is small, and both schemes' performance approaches that of the ideal schemes. From simulation, the performance of the best four-ring scheme is less than that of the best three-ring scheme, which supports our conjecture that $(n+1)$-ring schemes may not be better than n-ring schemes for $n \geq 3$. The network density and the maximum number of hops

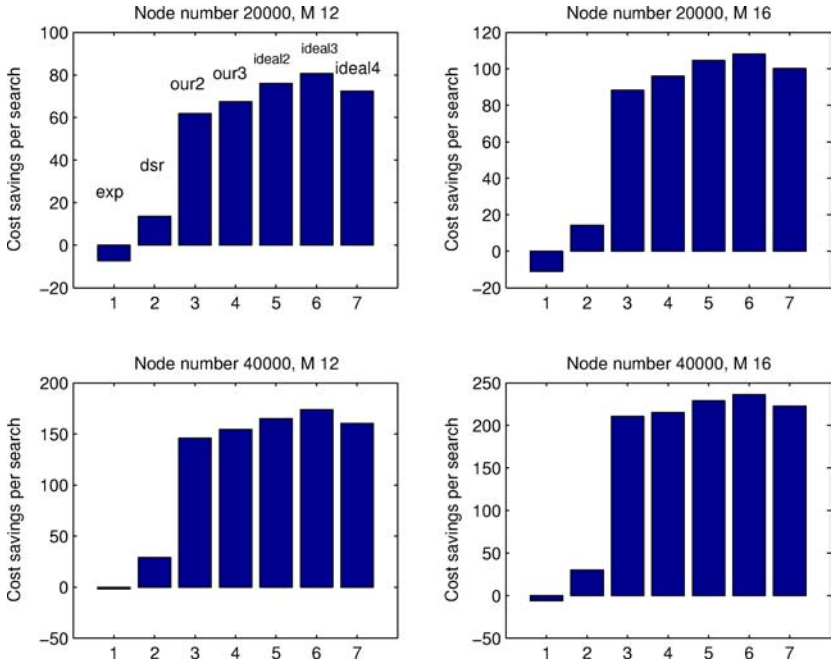


Figure 4. Cost savings per search for each scheme. From left to right labeled as 1 to 7: (1) expansion ring, (2) DSR, (3) our two-ring scheme, (4) our three-ring scheme, (5) ideal two-ring scheme, (6) ideal three-ring scheme and (7) ideal four-ring scheme. The y-axis indicates the number of packets saved per search.

M have an effect on the amount of the cost savings. However, they do not affect our above conclusions.

Earlier, we have claimed that all other schemes' latency is larger than the once-for-all scheme. Through simulations, we find that our two-ring and three-ring schemes have close cost performance and they have a higher latency than the once-for-all scheme with a percentage around 50–60%. The expansion ring scheme has around 120% higher latency, while the DSR scheme has only around 10% higher latency. The latency is not related to the network density. When the network density changes, N_i changes with the same scale, and from equation (4), L remains the same.

6.3. Performance comparison for small-scale networks with location knowledge

In this part, we compare the performance of different schemes in a small-scale network in which nodes have knowledge of their own locations x_0 as in figure 2. The total number of nodes N_T varies from 1000, 4000, 7000 to 10000. Nodes have different costs for target searching based on their locations. In figure 5, the x-axis is the different location x_0 of the investigated nodes. The y-axis is the cost of nodes at location x_0 applying different schemes of once-for-all, expansion ring, DSR, our two-ring scheme, the ideal two-ring scheme and the ideal three-ring scheme. Our scheme is to estimate N_i with the given location x_0 as $\tilde{N}_{i,x_0}$ and find the point where $\tilde{N}_{k,x_0} = \tilde{N}_{k+1,x_0}$ as the first attempt hop number. The second attempt hop number is chosen large enough to cover the entire network.

As can be seen from figure 5, our scheme performs consistently close to that of the ideal schemes, which indicates that our estimation matches reality quite well. DSR also performs consistently and is close to the once-for-all scheme. Nodes at different locations have much different costs when applying the expansion ring scheme. Some nodes may achieve less cost while other nodes may achieve more cost. Overall, the expansion ring's cost tends to be larger than that of the once-for-all scheme. When $x_0 = 1$, which means nodes are close to the center, the average cost decreases since more nodes are a relatively smaller number of hops away from the center.

As for latency, nodes that are close to the center usually need less time to cover the whole network and the resulting latency in finding the target is smaller.

6.4. Performance comparison for small-scale networks without location information

In this part, we compare the system cost savings of different schemes in a small-scale network without location information. Nodes have to apply a consistent parameter for a searching scheme instead of choosing different values for the first attempt.

In the right part of figure 6, we find that our estimated optimal point through equation (10), which is 13 for the network of 1000 nodes and 16 for the network of 2000 nodes, is quite close to the real optimal point. For this reason, the cost of our scheme is also close to that of the optimal scheme. Again, the expansion ring is the worst scheme and the DSR scheme achieves little cost savings.

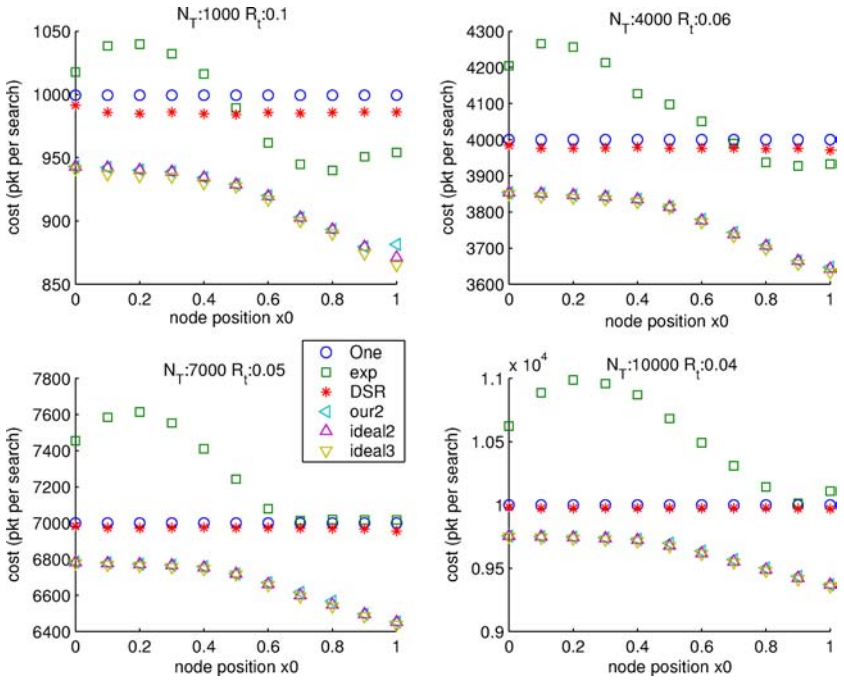


Figure 5. Cost of different schemes for nodes at different locations in small-scale networks. The X-axis indicates the location of the investigated nodes. The Y-axis indicates the cost of different schemes. Different network sizes of 1000, 4000, 7000 and 10000 nodes are simulated. The tested schemes are: the once-for-all scheme, the DSR scheme, the expansion ring scheme, our two-ring scheme, the ideal two-ring scheme and the ideal three-ring scheme.

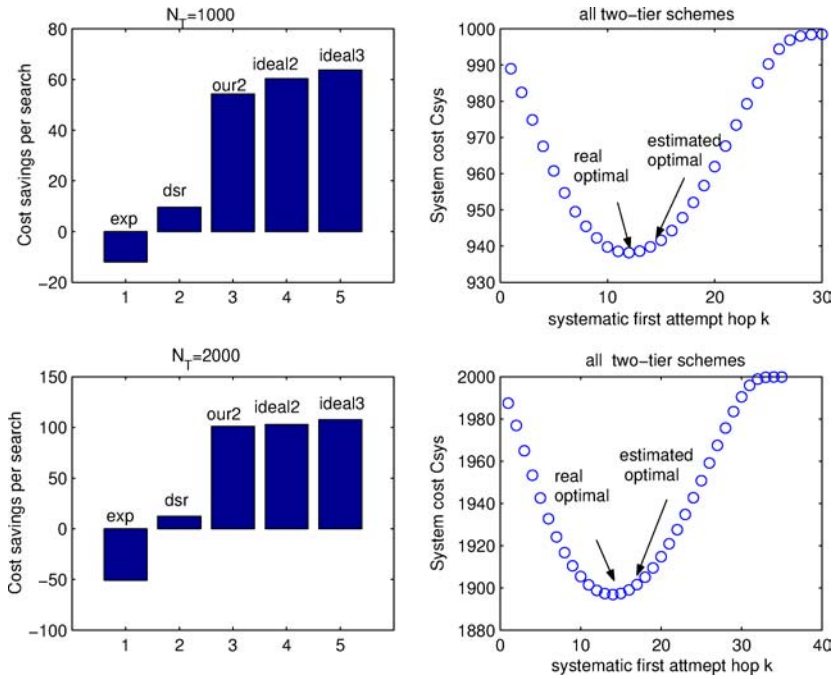


Figure 6. The cost savings of different schemes for small-scale networks. From left to right labeled as 1 to 5: (1) expansion ring, (2) DSR, (3) our two-ring scheme, (4) the ideal two-ring scheme, (5) the ideal three-ring scheme. The graph on the right shows the system cost for different k_{sys} of the two-ring scheme. Our estimated points are 13 for 1000 nodes and 16 for 2000 nodes, which are close to the true optimal points.

6.5. Summary

Based on our analysis for different network scales, we conclude the single target discovery problem with Table 2.

From analysis and simulation results, we find that, first, the cost saving percentage using an optimal n-ring scheme rather than the once-for-all scheme is less than 10% for most of the cases. Second, the cost saving percentage decreases when the number of nodes involved increases.[4] Third, the latency increases significantly when using an n-ring scheme instead of the basic once-for-all scheme. All this information implies that, for fast discovery, the best strategy is to flood just once. By applying the schemes presented in this paper, less than 10% of cost (lower in most cases) can be saved. Thus, even the optimal scheme may not be worth using, as there will be a substantial sacrifice in latency in exchange for an insignificant gain in cost savings. DSR and the arbitrary "expansion ring" scheme are not suitable for this target discovery problem due to their performance degradation in both cost and latency.

Table 2
Summary of the single target discovery problem.

Network scale	Flooding control	Metrics	Proposal	Parameters
Large	Hop-based	Cost	Two-ring	$\lceil \frac{M}{2} \rceil$,M
			Three-ring	$1,\lceil \frac{M}{2} \rceil$,M
		Latency	One-ring	M
Small	Location aware	Cost	Two-ring	$N_k = N_{k+1}$
	Location unaware	Cost	Two-ring	Valley of C_{sys}

[4] We conjecture that the percentage saving is on the order of $\sqrt{\frac{\log(N)}{N}}$.

7. Multi-target discovery

So far, we have studied the single target discovery problem, finding *one* target from a total of *one* target. In this section, we extend our study to the multi-target discovery problem, which is to find a certain number of targets from multiple targets. The existence of multiple targets makes the multi-ring searching scheme more promising in reducing searching cost. We will mainly look at the problem of finding at least *one* target out of *multiple* targets. We name this branch the one-out-of-m problem. The study of a more general problem of finding k out of m targets is similar, and here we only provide the conclusions for the one-out-of-m problem.

7.1. One-out-of-m

The only difference between this problem and the previous one is that there are a total of m equivalent targets distributed uniformly within a unit area. The question is the same: what is the optimal scheme to search this unit area to find at least one target using the minimum cost? In other words, how many searching attempts n should be performed and what should be the searching area set $\mathcal{A}^{(n)} = \{A_1, A_2, \ldots, A_n\}$ for these n searching attempts?

Here, we define cost as the total area that has been searched. This general assumption does not contradict the previous cost definition as the number of transmissions. In wireless networks, a node usually needs to forward packets for other nodes, and in order to search a certain area, the nodes within this area have to forward the queries. Thus, the number of query transmissions to search an area of A is proportional to A by

a constant coefficient determined by the forwarding mechanism such as flooding and gossiping. Also, by defining the cost directly as the searching area, we minimize the number of variables and simplify our analysis without loss of generality. The conclusions drawn from this definition can be specified for different applications simply by mapping the area to realistic application parameters.

To aid the expression, let us define a virtual 0th attempt search for the area of $A_0 = 0$. For the ith search attempt, the cost C_i is simply the cost summation of the first i attempts $C_i = \sum_{j=1}^{i} A_j$ The probability for one target to be outside the area A is $(1 - A)$, and the probability for all the m targets to be outside the area A is $(1 - A)^m$. In order to perform an ith search attempt and complete the task, all the m targets must be outside the area A_{i-1}, or else the task would have been completed earlier. Also, not all the m targets are outside the area A_i, otherwise the task will not end at the ith search attempt. Thus, the probability P_i for the task to be completed in the ith attempt is

$$\begin{aligned} P_i &= P\{\text{all targets outside } A_{i-1}, \\ &\quad \text{but not all targets outside } A_i\} \\ &= P\{\text{all targets outside } A_{i-1}\} \\ &\quad - P\{\text{all the targets are outside } A_i\} \\ &= (1 - A_{i-1})^m - (1 - A_i)^m \end{aligned} \tag{11}$$

The expected cost C^n for a general n-ring searching approach is

$$\begin{aligned} C^n &= \sum_{i=1}^{n} P_i C_i \\ &= \sum_{i=1}^{n} ((1 - A_{i-1})^m - (1 - A_i)^m) \left(\sum_{j=1}^{i} A_j \right) \\ &= \sum_{i=0}^{n-1} A_{i+1}(1 - A_i)^m \end{aligned} \tag{12}$$

The final equality above can be easily proven through mathematical induction.

7.2. Algorithms

Based on equation (12), we can perform a brute force search over the n-dimensional space $\mathcal{A}^{(n)} = \{A_1, \ldots, A_n\}$ to find the optimal solution $\mathcal{A}^{(n)}_{opt}$ to minimize C^n, given a specific n. However, this method is prohibitive since the computational requirement increases as $O((\frac{1}{\delta})^n)$, where δ is the brute force searching granularity. To reduce computations, we propose two heuristic solutions: a pre-calculated algorithm called Ring Splitting (RS) and an online algorithm called Online Ring Splitting (ORS).

RS is a greedy algorithm that starts from the one-ring searching scheme with $[A_0 = 0, A_1 = 1]$ and splits a ring if it provides the largest cost reduction until there are no possible choices to split a ring to achieve any more cost savings. The procedure is as follows.

1. Start with the ring [0, 1].
2. With an existing n-ring scheme, a given ring set of $\{[0, a_1], [a_1, a_2], \ldots, [a_{n-1}, 1]\}$ already exists. Check all these n rings and find out the candidates that can be split to further reduce the cost.
3. Terminate if there are no more candidates. Else, go to Step 4.
4. Pick the candidate that will reduce cost the most and split it. Go back to Step 2.

Whether a ring between $[A_k, A_{k+1}]$ should be split and become a candidate is determined as follows.

1. From equation (12), find out the cost difference D between the old n-ring scheme and the new $(n + 1)$-ring scheme when A_j is inserted between the ring $[A_k, A_{k+1}]$.

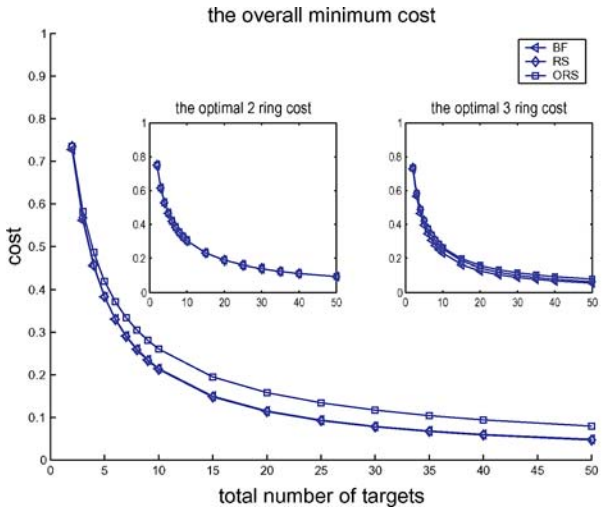
$$\begin{aligned} D &= C^n - C^{n+1} \\ &= A_{k+1}(1 - A_k)^m - (A_j(1 - A_k)^m + A_{k+1}(1 - A_j)^m) \end{aligned} \tag{13}$$

2. By solving $\frac{\partial D}{\partial A_j} = 0$, we have the potential splitting point A_j. Numerical methods are required to find A_j.
3. First, check if A_j is within $[A_k, A_{k+1}]$. Second, check if $D(A_j)$ is larger than zero. Only when both requirements are satisfied, should A_j be a ring splitting candidate for $[A_k, A_{k+1}]$.

ORS is derived from RS for online calculations. Unlike RS that calculates the entire searching area set in advance, ORS tries to split the remaining ring to achieve the lowest expected cost upon each failure. Due to the lack of global knowledge, ORS can only split the remaining ring; hence it performs slightly worse than RS. However, it requires even less computation than RS. There is only one computation for each additional searching attempt, and there is no wasted computation.

Figure 7 shows the optimal expected cost for each algorithm, where BF indicates the brute force method. The subplots show the optimal expected cost for the 2-ring and 3-ring schemes. The table shows the overall optimal cost by each algorithm and the respective computations. By comparing the results of the RS, ORS, and brute force approaches, we reach the following conclusions.

1. For each specific n, there exists an optimal solution. The overall optimal cost is reduced when n increases from 1. However, there exists a certain value n_0 such that the optimal overall cost will not decrease after this value. In the table of figure 7, BF finds its optimal at the 4th ring, while RS finds its optimal at the 6th ring and ORS finds its optimal at the 5th ring.
2. The optimal two-ring approach may reduce the cost dramatically, while the optimal three-ring approach may reduce cost around 2 to 5 more percent and it is already very close to the overall optimal. More searching attempts can only reduce the cost by a negligible amount of less than



An example of searching one out of 3 targets

scheme	optimal n	optimal cost	computations
BF	4	0.560852	165667502
RS	6	0.567105	9
ORS	5	0.581804	<4

Figure 7. The optimal expected cost for each solution and the optimal performance for 2-ring and 3-ring schemes. The cost is normalized by the cost of the once-for-all scheme. The x-axis indicates the targets available in the searching area. The y-axis indicates the minimum expected cost. Details about the behavior of different schemes for finding one out of three targets are shown in the table on the right.

1% and are unnecessary. This can be clearly seen from the subplots in figure 7.

3. RS performs close to BF while ORS requires even fewer computations with only a trivial 2–5% extra cost. The optimal cost saving ratio is $(1 - 0.38) \times 100\% = 62\%$ for searching a total of 5 targets and $(1 - 0.08) \times 100 = 92\%$ for 30 targets (see figure 7). When more targets are available, a smaller local searching area is preferred and the expected searching cost is smaller.

Unlike the single-target discovery problem where only insignificant cost savings can be obtained, a considerable amount of cost can be saved in the multi-target discovery problem. The comparison of our algorithm RS and the brute force method indicates that a simple 2-ring scheme can save almost as much searching cost as the optimal scheme as long as the first searching ring is properly chosen. Therefore, our next work is to implement RS in ad hoc networks, especially hop-based small-scale networks, and to compare its performance with DSR and EXP in terms of cost and latency.

8. Conclusions

In this paper, we extensively studied target discovery problems in wireless networks, including single target discovery and multi-target discovery. In summary, for the single target discovery problem, only insignificant cost can be saved even using the optimal searching scheme. For the multi-target discovery problem, a near maximum cost saving can be achieved using no more than 3 searching attempts. Algorithms to achieve this near optimal performance within 3 searching attempts are proposed.

In this paper we assume that targets are uniformly distributed in the searching area. This is the major reason why only insignificant searching cost can be saved for single target discovery. In reality, requested targets may be more likely to be located in the nearby area. How to determine the optimal multi-ring searching schemes for non-uniform target distribution discovery is of interest to us and will be studied in our future work.

In on-demand routing protocols, a source node finds the route to a destination either by finding the destination itself or by finding a route cache in intermediate nodes. Although this problem looks like a multi-target discovery problem, we cannot apply the conclusions in this paper directly. That is because we assume that targets are identical to each other in our study, while route caches may be stale and invalid. How to determine the optimal multi-ring searching schemes for target discovery with caching is another area of our future work.

References

[1] D. Braginsky and D. Estrin, Rumor routing algorithm for sensor networks, in: *Proc. International Conference on Distributed Computing Systems (ICDCS-22)* (2002).

[2] N. Bulusu, J. Heidemann and D. Estrin, Adaptive beacon placement, in: *Proceedings of the 21st International Conference on Distributed Computing Systems (ICDCS-21)*, Phoenix, Arizona, USA (April 2001) pp. 489–498.

[3] Z. Cheng and W. Heinzelman, Flooding strategy for target discovery in wireless networks, in: *Proc. of the 8th International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWIM 2003)* (Sept. 2003).

[4] Gnutella peer-to-peer file sharing system. http://www.gnutella.com.

[5] Y.-C. Hu and D.B. Johnson, Caching strategies in on-demand routing protocols for wireless networks, in: *Proc. ACM/IEEE MobiCom* (Aug. 2000) pp. 231–242.

[6] Y.-C. Hu and D.B. Johnson, Ensuring cache freshness in on-demand ad hoc network routing protocols, in: *Proc. POMC 2002 Workshop on Principles of Mobile Computing*, ACM, Toulouse, France (October 2002) pp. 25–30.

[7] C. Intanagonwiwat, R. Govindan and D. Estrin, Directed diffusion: A scalable and robust communication paradigm for sensor networks, *Mobile Computing and Networking* (2000) pp. 56–67.

[8] D.B. Johnson and D.A. Maltz, *Mobile Computing*, Chapter Dynamic source routing in ad hoc wireless networks, pp. 153–181. Kluwer Academic Publishers, Imielinski and Korth edition, 1996.

[9] M.K. Marina and S.R. Das, Performance of route caching strategies in dynamic source routing, in: *Proc. the Int'l Workshop on Wireless Networks and Mobile Computing (WNMC) in Conjunction with Int'l Conf. on Distributed Computing Systems (ICDCS)* (2001) pp. 425–432.

[10] D.L. Mills, Network Time Protocol (Version 3), RFC (Request For Comments) 1305 (March, 1992).

[11] T. Oates, M.V. Nagendra Prasad and V.R. Lesser, Cooperative information gathering: A distributed problem solving approach, Computer Science Technical Report 94-66-version 2 , University of Massachusetts, Amherst, MA.

[12] C. Perkins and E.M. Royer, Ad hoc on-demand distance vector routing, in: *Proceedings of IEEE WMCSA'99* (Feb. 1999) pp. 90–100.

[13] M. Roman, C.K. Hess, R. Cerqueira, A. Ranganathan, R.H. Campell and K. Nahstedt, Gaia: A middleware infrastructure to enable active spaces, *IEEE Pervasive Computing* (Oct-Dec, 2002) pp. 74–83.

[14] N. Sadagopan, B. Krishnamachari and A. Helmy, ACtive query forwarding in sensor networks (ACQUIRE), Computer Engineering Technical Report CENG 02-11 (2002).

[15] Sun Mircorsystems, *System and Netowrk Administration* (March, 1990).

[16] E. Woodrow and W. Heinzelman, SPIN-IT: A data centric routing protocol for image retrieval in wireless networks, *Proc. International Conference on Image Processing (ICIP '02)* (Sep, 2002).

[17] T. Wu, M. Malkin and D. Boneh, Building intrusion tolerant application. in: *Proc. of the 8th USENIX Security Symposium* (1999).

[18] Y. Xu, J. Heidemann and D. Estrin, Geography-informed energy conservation for ad hoc routing, in: *Proc. Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking*, Rome, Italy (July, 2001).

Zhao Cheng is a PHD. candidate in the Department of Electrical and Computer Engineering at the University of Rochester. He received a B.S. degree in Radio Engineering from Southeast University, China in 2000 and M.S. degree in Electrical and Computer Engineering from University of Rochester in 2003. His current research interests lie in the areas of sensor networks, quality of service (QoS) and reliability for mobile ad-hoc networks, and efficient discovery strategies for mobile ad-hoc networks.

Wendi Heinzelman is an assistant professor in the Department of Electrical and Computer Engineering at the University of Rochester. She received a B.S. degree in Electrical Engineering from Cornell University in 1995 and M.S. and Ph.D. degrees in Electrical Engineering and Computer Science from MIT in 1997 and 2000, respectively. Her current research interests lie in the areas of sensor networks, quality of service (QoS) and reliability for mobile ad-hoc networks, and multimedia communication. She is a member of Sigma Xi, the IEEE, and the ACM.