Combined TTL-Based Search Algorithm

***Abstract*—Expanding ring search and blocking expanding ring search are two prominent time-to-live value based search techniques for routes and resources in networks. Each uses a different technique to extend the search, with advantages in cost reduction that depend on the setting. In this paper, we introduce the Combined Expanding Ring Search which uses both techniques for extending the search in a way that guarantees expected costs at least as low as the lowest of the two. We show how to derive the search strategy with minimum expected costs in polynomial time, and in so doing introduce an important optimization to blocking expanding ring search. We further show how to convert an existing sequence of one type to a combined sequence in linear time, and how to derive the optimal sequence under delay constraints in polynomial time as well. In numerical evaluations of synthetic settings, we show that our optimization to blocking expanding ring search makes it far more efficient than previously considered, and consequently the combined search to far more efficient than either technique.**

I. Introduction

Mobile ad hoc networks are self-forming networks of mobile wireless communication devices that do not have a fixed infrastructure. Since they are characterized by dynamic topology changes, on demand routing protocols are especially suitable, because they limit overhead to what is needed for active routes rather than proactively maintaining routing tables by searching for routes as needed. Low cost search techniques are therefore essential to their performance.

In this paper, we present the Combined Expanding Ring Search (CERS) for routes in a network. CERS combines the techniques of two prominent search techniques–expanding ring search (ERS) [5, 8, 10] and blocking expanding ring search (BERS) [19, 21]. Both are flooding-based searches that limit the search extent by the number of hops a query is forwarded from the source. In ERS, a node assigns a query a time-to-live (TTL) value and forwards it to all of its neighbors, which decrement the TTL value and forward the query to their neighbors, and so on until the TTL value reaches zero, thus *flooding* the network with the request up to a limit. If the source does not receive a reply, it sends the query again with a larger TTL value, thus expanding the search in *rings* around it. In BERS, the source broadcasts the query once and stops the flooding when a path is found by broadcasting a chase packet up to all nodes that received the query. BERS eliminates the repetitive costs of ERS, but it incurs the additional cost of flooding the chase packet. CERS uses the best of both approaches in each search iteration, either initiating a new request from the source or continuing to flood from the last ring. The expected cost of CERS is guaranteed to be at least as low as the better of ERS and BERS.

The significance of our study is that we show how to derive the sequences that minimize the expected cost of using CERS. Most studies justify their techniques through simulations and qualitative analysis, but provide no performance guarantees and do not even guarantee that their techniques are better than a simple technique like flooding the entire network. This was especially problematic in the study of BERS, because they failed to note that the expected cost of their design is typically greater than the expected cost of the optimized ERS and can even be greater than the cost of full flooding. We, on the other hand, provide polynomial time algorithms to derive the optimal CERS sequence and guarantee that its expected cost will be no more than that of the optimal ERS and BERS strategies nor the cost of flooding. We utilize a simple but overlooked optimization to BERS that not only significantly reduces expected costs, but actually makes it more efficient than ERS in many cases, even when ERS is no better than flooding alone. This way, we make the most of a simple technique of limiting the number of hops a query is forwarded in order to efficiently search for routes in network.

The structure of this paper is as follows. Section II reviews related work, including related measures taken to optimize ERS and alternative ways to reduce search costs and routing overhead overall. The model and assumptions are presented in Section III. Section IV reviews ERS and BERS, including how to calculate the expected costs and times and a qualitative comparison. Section V introduces and analyzes CERS, presents our optimization to BERS, and shows how to derive the optimal CERS strategy in $O(N^2)$ time, where N is the maximum search range. Section VI presents a linear-time algorithm to optimize existing sequences of search ranges as knowledge of the network properties change. Section VII shows how to derive the optimal CERS strategy under delay constraints. Section VIII provides quantitative comparisons between the techniques. Section IX concludes the paper with a summary and discussion.

II. Related Work

Although the introduction focused on MANETs, expanding ring search and such techniques have been extensively analyzed for use in peer-to-peer and sensor networks as well [1, 3, 15, 27, 28].

Like ERS and BERS, most other search techniques for MANETs limit the search extent before flooding the entire network. LAR-1 [16] and PAR [29] limit the search to rectangular and elliptical regions, respectively. LAR-2 [16] and FRESH [11] forward requests to nodes closest to the target in space and time, respectively. Query localization (QL) [17] searches near the last known path. Each has its limitations. ERS and BERS cover more area than is needed. LAR-1, LAR-2, and PAR require knowledge of node location. FRESH depends on

the amount of contact between nodes. QL is only for route failure recovery.

Two variations of ERS are elastic ring search (ELRS) [26] and two-sided expanding ring search (BiERS) [25]. In ELRS, delays are deliberately inserted between search rounds to take advantage of changes in the topology. In BiERS, both nodes conduct expanding ring searches that meet midway between the two nodes. Like CERS, these are guaranteed to have lower expected costs than ERS, but their application is also limited. ELRS is for use when search delay is tolerated, and BiERS requires some prior coordination between the nodes.

The analysis in this paper is similar to that in [5], which uses dynamic programming to derive ERS sequences with minimum expected costs. Several other studies use closed-form expressions to optimize specific types of ERS sequences [8–10, 13, 14, 20], but closed-form expressions for the optimal sequence overall are difficult to derive. These studies assume that the probability distribution of the hopcount to the target is known. The deterministic and randomized strategies with the best average case competitive ratios when this distribution is not known are derived in [4] and [7], respectively. The same results were obtained in [12] with respect to the time needed to complete an expanding ring search. Optimal strategies under constraints on the expected search time are derived in [6].

The study of BERS has focused on pause times rather than search sequences. In BERS [19], each node waits long enough to receive a chase packet before forwarding a query. BERS* [21] cuts the wait time in half in order to reduce the overall search delay, but the query is forwarded one more hop than necessary as a result. The overhead can be reduced by initiating the chase packet from the target [18, 24, 30], but this does not guarantee any fundamental difference in expected costs. ERDA [2] only pauses after the query is outside some neighborhood of the source in order to reduce search delay. Other techniques transmit the query at a slower speed than the chase packet [12, 22], but this is effectively the same as delaying packet transmissions. Gargano, et al. [12] determine the ratio of transmission speeds with the best worst-case search time. None of these studies provide guarantees about the expected search costs, and only [12] gives any theoretical analysis of their strategy. The cost/time tradeoff of BERS is studied in [23].

III. Preliminaries

Our design and analysis of CERS is similar to that of ERS and BERS [5, 8, 10, 19, 21]. A maximum limit N is placed on the search extent, and the search is terminated when a query is forwarded up to that extent. This limit can be any value, such as the network diameter, the maximum number of hops to the target, or at worst the number of nodes in the network. The assumption is that the target is guaranteed to be found by a query up to this extent, but changing this assumption does not affect the results. The search extents are defined by the number of hops a query is forwarded from the source and are also referred to as rings. Associated with the range of search extents are a cost function $C(i)$, a probability mass function (PMF) $f(i)$ of the minimum hop count to the destination, and the corresponding cumulative distribution function (CDF) $F(i)$. The cost function reflects the effects of broadcast flooding on network performance, such as the power supply of individual devices, bandwidth, and packet loss. We further assume that the marginal cost of forwarding a message from ring i to ring j is $C(j)-C(i)$. The cost function and PMF can be determined in advance according to the network parameters, such as node density and spatial distribution, or estimated by sampling and distributed learning. The values of $C(0)$ and $F(0)$ are defined as 0.

We associate the time to forward a message one hop with a constant unit of time, which is a standard assumption in the design of routing protocols. The time to forward a packet one hop includes the time to transmit, receive, and process the packet, as well as possible backoff and queuing delays. Although these delays may vary with network conditions, we go with the assumption that they are the same on average.

Some equations refer to the conditional PMF and CDF of the minimum hop count when it is known to be greater than i, denoted $f(j|i)$ and $F(j|i)$, respectively. They can be calculated as follows:

$$f(j|i) = \begin{cases} \frac{f(j)}{1-F(i)} & i < j \leq N \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$F(j|i) = \begin{cases} \frac{F(j)-F(i)}{1-F(i)} & i < j \leq N \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

IV. ERS and BERS

Expanding ring search and blocking expanding ring search are used by routing protocols and network applications for route, service, and resource discovery. In both techniques, the search iteratively expands outward from the source according to limits placed on the number of hops from the source. They differ in how the search is extended to increasing limits. In this section, we formally define both techniques, discuss their practical implementation, and analyze the expected cost, expected time, and worst-case time of both techniques. In both search techniques, the destination must send a reply to the source in order to establish a connection. Even after the source receives a reply, other nodes may still be continuing the search. We only consider the total cost of flooding and the time until a connection is established. We do not consider the additional cost of sending a reply because it is fixed, regardless of the method used. We do not consider the time until all nodes cease to forward requests after the reply is received by the source because it is irrelevant once the destination is found.

A. Expanding Ring Search

In expanding ring search (ERS), the source node iteratively floods a route request to increasing hop limits until the target is found. This is implemented by assigning a route request a time-to-live (TTL) value d_i in iteration i and broadcasting it to all neighbors. Every node that receives a request with a TTL value greater than one for the first time decrements

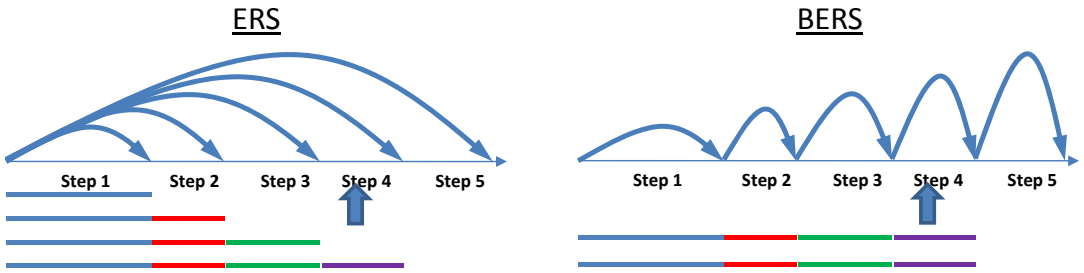

Fig. 1. Expected cost structure with ERS (left) and BERS (right).

the TTL value and rebroadcasts the query. The source waits $2d_i$ time units, and if a route reply is not received in that time, then it broadcasts a new query with a TTL value $d_{i+1} > d_i$. A search strategy is a sequence of m increasing TTL values, $\{d_1, \ldots, d_m = N\}$. The expected cost of using such a sequence is:

$$\sum_{i=1}^{m} C(d_i)(1 - F(d_{i-1})) \tag{3}$$

The optimal ERS strategy is the sequence that minimizes this formula, and it can be derived in polynomial time using standard dynamic programming techniques [5].

Each iteration i must be long enough for the query to be forwarded d_i hops away and for a reply to be sent back from this extent if the target is found. The worst-case time of an ERS strategy equals two times the sum of the TTL values. The expected time can be calculated using the following formula. The first term is for the expected time to forward the query to the ring in which the target is found and to receive a reply, and the second term is for the expected time for waiting between search rounds.

$$2\sum_{i=1}^{N} if(i) + 2\sum_{i=1}^{m-1} d_i(1 - F(d_i)) \tag{4}$$

B. Blocking Expanding Ring Search

In the standard definition of BERS, the source broadcasts the query only once and each node that receives it pauses two time units times its distance from the source before forwarding it to the next ring. This pause is also called "blocking". In this time, if a route is found, a reply is sent to the source, and it in turn floods a chase packet up to the furthest ring the query reached to cancel the flooding. This can be generalized by defining a strategy $\{d_1, \ldots, d_m = N\}$, $d_{i+1} > d_i \ \forall 1 \leq i < m$, and only blocking at the rings in this sequence. To implement BERS in its most general form, the query must contain the sequence of search extents and a counter of how many hops it is from the source. Every node that receives the query increments the counter and pauses $2d_i$ time steps before forwarding the query if the counter equals any d_i in the search sequence. The source floods a chase packet with TTL value d_i when it receives a reply that is between d_{i-1} and d_i hops away, except when $d_i = N$.

The expected cost of using a strategy $\{d_1, \ldots, d_m = N\}$ can be calculated using the following formula. The first term is the sum of the marginal cost of flooding from ring d_{i-1} to ring d_i times the probability that search will be extended beyond ring d_{i-1}. This is multiplied by two to cover the cost of forwarding the query and the chase packet when the target is found. Note that a chase packet does not need to be sent when the target is more than d_{m-1} hops away, since the flooding will terminate on its own. Therefore, the cost of sending a chase packet to ring d_m is excluded by subtracting $C(d_m)(1 - F(d_{m-1}))$.

$$2\sum_{i=1}^{m}(C(d_i) - C(d_{i-1}))(1 - F(d_{i-1})) - C(d_m)(1 - F(d_{m-1})) \tag{5}$$

The maximum time of using such a strategy is two times the sum of all values in the sequence, because time is needed for blocking at each of the rings, except for the last, plus time to send the query the furthest extent and receive a reply in the worst-case. The expected time can be calculated using (4). In this case, the the second term is for the expected time for blocking.

C. Methods Comparison

The differences between ERS and BERS are illustrated in Figure 1. In ERS, if the target is found within the i^{th} iteration, then the marginal cost $C(d_j) - C(d_{j-1})$, $(1 \leq j \leq i)$ is incurred $i - j + 1$ times. In BERS, if the target is found in the i^{th} iteration or later, then the marginal cost $C(i) - C(i-1)$ is incurred twice — the first time as part of the marginal flooding, and the second time when sending the cancellation message. The same amount of pause time is required after flooding up to a particular ring and either initiating a new query, as in ERS, or blocking at that ring, as in BERS.

The advantage of BERS over ERS is that the source does not need to re-initiate the search. However, it incurs the cost of the cancellation flood, which may be more costly than the repeated re-initiation of the search. Similarly, ERS floods the same sections multiple times; however, the section in which the destination is found is flooded only once.

V. CERS

Recognizing the specific advantages of ERS and BERS, we present a combined method, Combined Expanding Ring

Search (CERS), that realizes the benefits of both. In the combined method, the source iteratively issues either an ERS-type query or a BERS-type query. An ERS-type query is flooded up to some hop distance r from the source without pausing between rings, where $r' < r \leq N$ and r' is the maximum extent of the previous query. The BERS-type query is a blocking expanding ring search with a sequence $r_1, \ldots, r_l$, where $r' < r_1 < N$ and $r_l \leq N$. The idea behind CERS is that the search extent is extended using the method that minimizes the expected overall cost. In the remainder of the section, we show how to derive the optimal CERS strategy, discuss some important properties of CERS, and compare CERS to ERS and BERS.

A. Indifference point

Our design of CERS incorporates a major improvement to BERS that was not considered in previous studies. In the standard definition of BERS, a chase packet is always sent when a route is found, even in the last ring. This would be the optimal strategy if the maximum search extent was infinite, which can be proven as follows: The contribution of any pair of successive values d_i and d_{i+1} in a general BERS sequence to the expected cost is

$$2(C(d_{i+1}) - C(d_i))(1 - F(d_i)) \\ = 2 \sum_{k=d_i}^{d_{i+1}-1} (C(k+1) - C(k))(1 - F(d_i))$$

On the other hand, the contribution of the sequence $d_i, d_i + 1, \ldots, d_{i+1}$, in which blocking occurs at every ring between d_i and d_{i+1}, to the expected cost would be

$$2 \sum_{k=d_i}^{d_{i+1}-1} (C(k+1) - C(k))(1 - F(k)) \\ \leq 2 \sum_{k=d_i}^{d_{i+1}-1} (C(k+1) - C(k))(1 - F(d_i))$$

This is because $1 - F(k) < 1 - F(d_i)$ for any value $k > d_i$. Therefore, it is more cost effective to replace any subsequence d_i, d_{i+1} with the sequence $d_i, d_i + 1, \ldots, d_{i+1}$.

In our generalization of BERS, the source does not broadcast a cancel message to the last ring in the search sequence since the search terminates on its own. Therefore, at some point it is less costly to forward the query to the last ring without blocking inbetween than to broadcast a cancel message if a path is found. This point—the *indifference point* — is precisely the point D at which $C(N) - C(D) < C(D)$. Accordingly, the optimal strategy is to block at each ring up to the indifference point, and from there onwards to forward the query without blocking.

Figure 2 highlights the importance of including the indifference point in a BERS strategy. It compares the expected costs of standard BERS to that of BERS with the indifference point in a synthetic scenario. It further includes the optimal search cost, which is the expected cost of flooding up to the target, as a theoretical lower bound, and the expected cost of optimized ERS. The settings are $N = 20$, $F(i) = (i/20)^n$, for the range of values n on the x-axis, and $C(i) = (i/20)^{0.5}$. According to these settings, the cost of flooding is 1, which should therefore be the maximum cost of any search technique. This figure shows that BERS without the indifference point is more costly than ERS and even flooding in some cases. However, it is much less costly with the indifference point, even less than ERS in most cases, even when ERS is no less costly than flooding. Until now, these have been overlooked yet highly significantly properties of BERS.

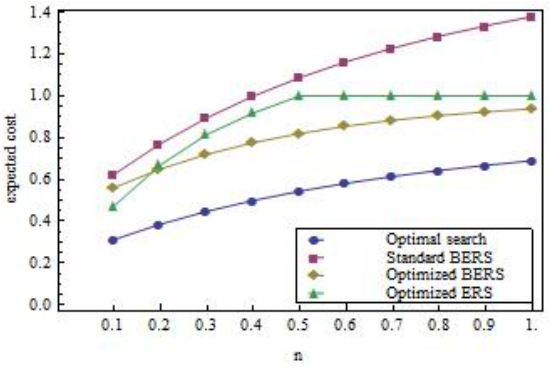

Fig. 2. Comparison of expected costs of BERS, ERS, and optimal search

B. Analysis

To simplify the analysis, we define a CERS sequence differently than the protocol described at the beginning of this section. A CERS sequence is formally defined as $(d_1, s_1), ..., (d_m, s_m)$, $d_{i+1} > d_i$, $\forall 1 \leq i < m$, $d_m = N$, where $s_i \in \{E, B\}$ is the search method to be used if the target has not been found in any search up to d_i. Successive pairs $(d_i, E), (d_{i+1}, s_{i+1})$ indicate that the source is to flood the network up to an extent d_{i+1}. Successive pairs $(d_i, B), (d_{i+1}, s_{i+1})$ indicate that the nodes at distance d_i from the source pause for $2d_i$ time units and resume flooding up to distance d_{i+1} if a cancellation message is not received. A cancellation message is sent to distance d_i whenever $s_i = B$. The pair (d_0, s_0) is always defined as $(0, E)$, which is used to indicate the source first floods to d_1 without pausing.

In this definition of a CERS sequence, all the rings at which blocking occurs are unrolled. This type of sequence corresponds to the queries the source issues as follows. A pair (d_i, E) followed by a pair (d_{i+1}, E) corresponds to an ERS-type query up to range d_{i+1}. A subsequence $(d_i, E), \ldots, (d_j, E)$ in which $j > i + 1$ and $s_k = B$ for all $i < k < j$ corresponds to a BERS-type query with sequence $d_{i+1}, \ldots, d_j$.

The expected cost of a CERS sequence can be calculated with the following formula.

$$\sum_{i=1}^{m} ((C(d_i) - C(d_{i-1}) \cdot \mathbf{1}_{s_{i-1}=B})(1 - F(d_{i-1}))$$

$$+ C(d_i)(F(d_i) - F(d_{i-1})) \cdot \mathbf{1}_{s_i=B}) \quad (6)$$

The expression $((C(d_i) - C(d_{i-1}) \cdot \mathbf{1}_{s_{i-1}=B})(1 - F(d_{i-1}))$ is the cost of forwarding the query up to ring d_i, depending on s_{i-1}, which is incurred with probability $1 - F(d_{i-1})$. The expression $C(d_i)(F(d_i) - F(d_{i-1})) \cdot \mathbf{1}_{s_i=B}$ is the cost of sending a cancellation message if blocking occurs at ring d_i and a path whose length is in the range $[d_{i-1}+1, d_i]$ is found. The expected time and maximum time can be calculated the same way it is calculated for ERS and BERS.

The optimal CERS sequence can be derived in polynomial time by solving the following dynamic programming equations. The terms $V(i, E)$ and $V(i, B)$ are the minimum expected costs of continuing the search using an E-step and B-step, respectively, when the target was not found in a prior query up hop count i.

$$\begin{aligned}
V(N, E) &= 0 \\
V(i, E) &= \min_{i<j\leq N}[C(j) + \min\{(1 - F(j|i))V(j, E), \\
&\qquad C(j)F(j|i) \cdot \mathbf{1}_{j<N} + (1 - F(j|i))V(j, B)\}] \\
V(N, B) &= 0 \\
V(i, B) &= \min_{i<j\leq N}[C(j) - C(i) + \min\{(1 - F(j|i))V(j, E), \\
&\qquad C(j)F(j|i) \cdot \mathbf{1}_{j<N} + (1 - F(j|i))V(j, B)\}] \qquad (7)
\end{aligned}$$

Both $V(N, E)$ and $V(N, B)$ equal 0 because the search ends once a query has been forwarded to ring N. For all other i, $0 \leq i < N$, $V(i, E)$ and $V(i, B)$ equal the minimum expected cost of continuing the search over all j, $i < j \leq N$. This is the cost of extending the search up to ring j ($C(j)$ for $V(i, E)$ and $C(j) - C(i)$ for $V(i, B)$) plus the expected cost of continuing the search from j onwards with either an E-step or a B-step. If the search is continued with an E-step, then the expected cost $V_E(j)$ is incurred if the target is not found (with probability $1 - F(j|i)$). If the search is continued with a B-step, then the cost $C(j)$ is incurred for sending a cancel message if a path is found (with probability $F(j|i) \cdot \mathbf{1}_{j<N}$), and the expected cost $V(j, B)$ is incurred for forwarding the message if a path is not found (with probability $1 - F(j|i)$).

These equations can be solved in $O(N^2)$ time using recursive functions and memoization, a technique to store results of previous function calls, or iteratively using a procedure like the one in Figure 3. The array value $V[i, s]$ corresponds to the value of $V(i, s)$, $\forall 0 \leq i \leq N \wedge s \in E, B$, and the array value $S[i, s]$ corresponds to the pair (j, s'), $i < j \leq N \wedge s' \in E, B$ that minimizes $V(i, s)$. In Line 1, $V[N, E]$ and $V[N, B]$ are set to 0. The values of $V[i, E]$, $V[i, B]$, $S[i, E]$, and $S[i, B]$, $0 \leq i < N$, are solved for in descending order of i in Lines 2-21. This is done by repeatedly replacing $V[i, E]$, $V[i, B]$, $S[i, E]$, and $S[i, B]$ with the best assignment found so far, beginning with the values corresponding to a query up to N (Lines 3 and 4). The asymptotic runtime of this procedure is $O(N^2)$, because the j loop is executed up to N times in each iteration of the i loop.

```
1:  V[N, E] ← 0; V[N, B] ← 0
2:  for i = N − 1 downto 0 do
3:      V[i, E] ← C(N); S[i, E] ← (N, E)
4:      V[i, B] ← C(N) − C(i); S[i, B] ← (N, E)
5:      for j = i + 1 to N − 1 do
6:          contE ← ((1−F(j))/(1−F(i))) V[j, E]
7:          contB ← C(j)((F(j)−F(i))/(1−F(i))) + ((1−F(j))/(1−F(i))) V[j, B]
8:          if C(j) + contE < V[i, E] then
9:              V[i, E] ← C(j) + contE; S[i, E] ← (j, E)
10:         end if
11:         if C(j) + contB < V[i, E] then
12:             V[i, E] ← C(j) + contB; S[i, E] ← (j, B)
13:         end if
14:         if C(j) − C(i) + contE < V[i, B] then
15:             V[i, B] ← C(j) − C(i) + contE; S[i, B] ← (j, E)
16:         end if
17:         if C(j) − C(i) + contB < V[i, B] then
18:             V[i, B] ← C(j) − C(i) + contB; S[i, B] ← (j, B)
19:         end if
20:     end for
21: end for
```

Fig. 3. Iterative algorithm to derive the optimal CERS strategy

C. CERS vs. ERS and BERS

The dominance of CERS over ERS and BERS is unquestionable, as the latter two are specific cases of the first — in the worst case, the CERS algorithm will produce a sequence which is equal to the best among the two. Proposition 1 suggests that for the degenerate cases in which $N \leq 2$, the CERS algorithm will produce exactly the best of the two.

Proposition 1. *For the case of $N \leq 2$, the expected cost of using the optimal CERS sequence is equal to the minimum among ERS and BERS (i.e., the CERS cannot improve the best between ERS and BERS for $D \leq 2$.*

Proof. For the case where $N = 1$, all three methods will be using the same sequence. For the case $N = 2$, the possible options are either go directly to distance 2 (in which case it is identical to ERS) or go to 1 and then to 2. Going to 1 is always direct (either in ERS or BERS) and going to 2 can be done via ERS or BERS, so CERS must always be identical to the optimal ERS or BERS scheme. □

VI. Adapting ERS and BERS to CERS

While an optimal CERS sequence can be produced using the algorithm in Figure 3, the computational complexity is not negligible. In real-life settings, where nodes are inherently limited computationally- and energy-wise, simpler solutions ought to be used. For this purpose, we present in this section an analysis of the tradeoffs associated with replacing ERS with BERS steps (and vice-versa), resulting in an efficient method to modify CERS sequences when assumptions about the hop count distribution change. The idea, then, would be for the nodes to start with the optimal CERS sequence, then modify

it as the assumptions change, and derive new sequences only when there are major changes to the initial assumptions.

At the foundation of the method is a simple rule for adjusting any ERS or BERS sequence to a CERS sequence, stated in Theorem 2.

Theorem 2. *For any search sequence S, (a) replacing a step (d_i, E) with (d_i, B) will necessarily improve the overall expected cost if $F(d_i|d_{i-1}) < \frac{1}{2}$, and (b) replacing a step (d_i, B) with (d_i, E) will necessarily improve the overall expected cost if $F(d_i|d_{i-1}) > \frac{1}{2}$.*

Proof. Let $J(S)$ be the expected cost of continuing the search using a sequence $S = (d_i, s_i), \ldots, (d_m, s_m)$, such that $J((d_i, E), S') = (1 - F(d_i|d_{i-1}))(C(d_{i+1}) + J(S'))$ and $J((d_i, B), S') = F(d_i|d_{i-1})C(d_i) + (1 - F(d_i|d_{i-1}))(C(d_{i+1}) - C(d_i) + J(S'))$, where $S' = (d_{i+1}, s_{i+1}), \ldots, (d_m, s_m)$ and $J((d_m, s_m)) = 0$. The difference between $J((d_i, E), S')$ and $J((d_i, B), S')$ is $(1 - F(d_i|d_{i-1}))C(d_i) - F(d_i|d_{i-1})C(d_i)$. Since $C(d_i)$ is positive, this expression is positive when $F(d_i|d_{i-1}) < \frac{1}{2}$, implying that it is more efficient to set $s_i = B$, and negative otherwise. □

The intuition behind this result is as follows. If $F(d_i|d_{i-1}) > \frac{1}{2}$, then there is a greater probability of incurring the cost $C(d_i)$ to cancel the flooding when $s_i = B$ than incurring the additional cost $C(d_i)$ to resend the query from the source when $s_i = E$.

Based on Theorem 2, we can propose the following algorithm for improving any sequence S. The algorithm makes one pass over the sequence from $i \leftarrow 1$ to $m - 1$. If $F(d_i|d_{i-1}) < \frac{1}{2}$, then set s_i to B; otherwise, set s_i to E. The following theorem proves that this algorithm will set all of the s_i to the optimal values for the corresponding values of d_i.

Theorem 3. *Given a search sequence $S = \{(d_1, s_1), \ldots, (d_m, s_m)\}$, the CERS adaptation algorithm will result in a sequence $S' = \{(d_1, s'_1), \ldots, (d_m, s'_m)\}$ such that the expected cost of S' is less than or equal to the expected cost of S'', $\forall S'' = \{(d_1, s''_1), \ldots, (d_m, s''_m)\}$.*

Proof. Since the replacement is based only on the probabilities and not on the values s_i, then once replacing a given value s_i, the decision will not change based on other replacements. □

This algorithm is particularly useful in settings where the optimal ERS or BERS has already been pre-calculated, as it enables a performance improvement with a very limited computational load. Therefore, in systems that are already based on ERS or BERS sequences, an additional computational layer can simply be added without further complicating the core calculations.

VII. Delay Constraints

In many settings, the search process is constrained by a maximum allowable delay from the beginning of the search until the searcher receives a reply. This delay is usually defined by an application-dependent service-level agreement (SLA). In such cases, the optimal solution must guarantee that even if the sequence is fully executed, the overall delay does not exceed the specified value. In this section, we show how to derive the optimal ERS, BERS, and CERS strategies under delay constraints. Throughout the discussion, the delay constraint is denoted T. The search sequence for delay-constrained CERS is the same as for CERS without delay constraints. Because of this, Theorem 2 does not change when CERS is subject to delay constraints, and therefore all the results of Section VI apply here as well.

A. ERS

The optimal delay-constrained ERS strategy can be derived using the dynamic programming equations below. The term $V(i, t)$ is the expected cost of continuing the search given the last TTL value used, i, and the time t the current query is to be conducted at. The next TTL value, j, can be either N, thus terminating the search, or in the range $[i+1, N-1]$. A query with j will terminate at time $t+2j$. When $j < N$, at least $2N$ time units must be left over for a final query with TTL value N. That is, j must satisfy the inequality $t + 2j + 2N \leq T$, so j is chosen from the set $Q = \{i+1, \ldots, \min\{(T-t)/2 - N, N\}, N\}$.

$$V(N, t) = 0$$
$$V(i, t) = \min_{j \in Q}\{C(j) + (1 - F(j|i))V(j, t + 2i)\} \quad (8)$$

B. BERS

The optimal delay-constrained BERS strategy can be derived using a set of dynamic programming equations similar to (8). As in ERS, the next ring at which blocking occurs, j, can either be N, thus terminating the search, or in the range $[i+1, N-1]$. When $j = N$, the query will terminate at time $t + (N - i) + N$, which is the time to forward to query to ring N and send a reply to the source. When $j < N$, the query will terminate at time $t + (j - i) + 2j$, which includes the time to block at ring j. In this case, at least $(N - j) + N$ time units must be left over to complete the search. That is, j must satisfy the inequality $t + 2j + 2N - i \leq T$, so j is chosen from the set $Q = \{i+1, \ldots, \min\{(T-t+i)/2 - N, N\}, N\}$.

$$V(N, t) = 0$$
$$V(i, t) = \min_{j \in Q}\{(C(j) - C(i)) + C(j)F(j|i) \cdot \mathbf{1}_{j<N} + (1 - F(j|i))V(j, t + (2 + \mathbf{1}_{j<N})j - i)\} \quad (9)$$

C. CERS

The optimal delay-constrained CERS strategy can be derived using the set of dynamic programming equations below, which combines principles from the solutions for the optimal CERS and delay-constrained ERS and BERS strategies. For simplicity, the next step j is chosen from the set $[i+1, N]$, which may lead to an invalid solution. For this purpose, an additional condition is included that whenever $t > T$, the

$V[N, E|B, 0..T] \leftarrow 0$
$V[0..N-1, E|B, T] \leftarrow \infty$
$V[0..N, E|B, T+1..T+3N] \leftarrow \infty$
for $t = T-1$ downto 0 **do**
  for $i = 0$ to $N-1$ **do**
    $V[i, E, t] \leftarrow C(N) + V[N, E, t+2N]$
    $S[i, E, t] \leftarrow (N, E)$
    $V[i, B, t] \leftarrow C(N) - C(i) + V[N, E, t+2N-i]$
    $S[i, B, t] \leftarrow (N, E)$
    for $j = i+1$ to $N-1$ **do**
      $contEE \leftarrow C(j) + \frac{1-F(j)}{1-F(i)} V[j, E, t+2j]$
      if $contEE < V[i, E, t]$ **then**
        $V[i, E, t] \leftarrow contEE$; $S[i, E, t] \leftarrow (j, E)$
      end if
      $contEB \leftarrow C(j) + C(j)\frac{F(j)-F(i)}{1-F(i)} + \frac{1-F(j)}{1-F(i)} V[j, B, t+3j]$
      if $contEB < V[i, E, t]$ **then**
        $V[i, E, t] \leftarrow contEB$; $S[i, E, t] \leftarrow (j, B)$
      end if
      $contBE \leftarrow C(j) - C(i) + \frac{1-F(j)}{1-F(i)} V[j, E, t+2j-i]$
      if $contBE < V[i, B, t]$ **then**
        $V[i, B, t] \leftarrow contBE$; $S[i, B, t] \leftarrow (j, E)$
      end if
      $contBB \leftarrow C(j) - C(i) + C(j)\frac{F(j)-F(i)}{1-F(i)} + \frac{1-F(j)}{1-F(i)} V[j, B, t+3j-i]$
      if $contBB < V[i, B, t]$ **then**
        $V[i, B, t] \leftarrow contBB$; $S[i, B, t] \leftarrow (j, B)$
      end if
    end for
  end for
end for

Fig. 4. Iterative algorithm to derive the optimal delay-constrained CERS strategy

expected cost of continuing the search is ∞. The equations below are specifically for the case when $t \leq N$.

$$
\begin{aligned}
&V(N, E, t) = 0 \\
&V(i, E, t) = \min_{i<j\leq N} [C(j) + \min\{(1 - F(j|i))V(j, E, t+2j), \\
&C(j)F(j|i) \cdot \mathbf{1}_{j<N} + (1 - F(j|i))V(j, B, t + (2 + \mathbf{1}_{j<N})j)\}] \\
&V(N, B, t) = 0 \\
&V(i, B, t) = \min_{i<j\leq N} [C(j) - C(i) \\
&+ \min\{(1 - F(j|i))V(j, E, t+2j-i), \\
&C(j)F(j|i) \cdot \mathbf{1}_{j<N} + (1 - F(j|i))V(j, B, t+3j-i)\}]
\end{aligned}
\tag{10}
$$

Just as in Section V-B, these equations can be solved in $O(N^2T)$ time using recursive functions and memoization or iteratively using a procedure like the one in Figure 4. This algorithm utilizes a three-dimensional array $V[0..N, E|B, 0..T+3N]$ that corresponds to $V(i, E|B, t)$. The size of the third dimension is $T+3N$ to account for all the cases when $t > T$. The size could be reduced to T by including if statements any time V is referenced to substitute its value with ∞ if $t > T$, but this will also increase the runtime.

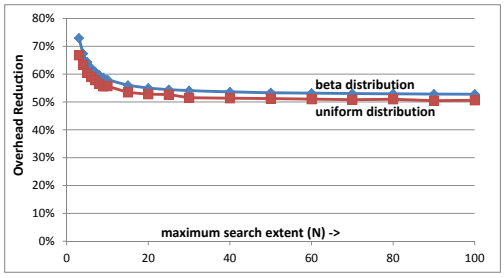

Fig. 5. OBERS vs. BERS with no delay constraint: average percentage of overhead reduction as a function of the maximum search extent N, for different generating distribution functions (uniform,beta).

VIII. Performance Numerical Illustration

In this section, we report the results of extensive numerical evaluation of BERS optimized with the indifference point BERS (OBERS) and CERS compared to ERS and BERS. The purpose of this numerical evaluation is primarily demonstrative, aiming to highlight the effect of the different model parameters over the performance achieved. The superiority of OBERS over BERS and of CERS over ERS, BERS and OBERS are non-questionable as these are specific cases—in the worst case OBERS will return a BERS sequence and CERS will return a ERS or OBERS sequence.

The numerical evaluation is based on synthetic settings that were generated using two probability distribution generating functions: uniform and beta. For the uniform case, we randomly drew the cost and probability at each distance D ($D \leq N$) from the interval $(0, 1)$ and normalized their values for each function. For each generated setting in the beta case, we first randomly drew the two parameters α and β from the range $(0.001, 10)$, set the cost at each distance D ($D \leq N$) to the cumulative value of a beta distribution with parameters α and β at D/N, and did the same for the CDF. For each tested condition (i.e., different values of N) and generating distribution function (uniform,beta) we generated 1000 settings according to the above guidelines. For each setting we derived its ERS, BERS, OBERS, CERS and adjusted ERS (i.e., based on local transitions to BERS steps when starting from the optimal ERS sequence) sequences. In addition, we calculated the theoretical minimal possible expected cost of each given setting, using the formula $\sum_{i=1}^{N} if(i)$. This is the expected cost of flooding up to exactly the number of hops to the target if it was known *a priori*.

Let $cost^{\mathcal{P}}(S)$ be the expected cost with method

$$\mathcal{P} \in CERS, ERS, BERS, OBERS, Adjusted, Theoretical$$

for a setting S. The method $\mathcal{P} = Adjusted$ represents the adaptation of an optimal ERS sequence through local changes of ERS steps to BERS steps, as discussed in Section VI. The cost $cost^{Theoretical}(S)$ is the theoretical (inachievable) minimum, hence will be used as a lower bound for comparison. The

difference $cost^{\mathcal{P}}(S) - cost^{Theoretical}(S)$ is thus the overhead of method $\mathcal{P}$ and only that amount can actually be improved. Therefore, when comparing a method $\mathcal{P}_i$ with method $\mathcal{P}_j$, such that $cost^{\mathcal{P}_i}(S) < cost^{\mathcal{P}_j}(S)$ an appropriate measure for the improvement achieved is by how much we managed to reduce the cost overhead, i.e., the cost beyond $cost^{Theoretical}(S)$. Therefore, our primary measure in the numerical evaluation, when comparing two methods $\mathcal{P}_i$ and $\mathcal{P}_j$ is the percentage of reduction in the cost overhead incurred when using $\mathcal{P}_i$ compared to when using $\mathcal{P}_j$. Formally, this is calculated by: $\frac{cost^{\mathcal{P}_j}(S) - cost^{\mathcal{P}_i}(S)}{cost^{\mathcal{P}_j}(S) - cost^{theoretical}(S)}$.

Figure 5 compares $BERS$ and its improvement using the indifference point as suggested in the paper, $OBERS$. It depicts the average percentage of overhead reduction achieved by using $OBERS$ rather than $BERS$, as a function of the maximum search extent N, for the two generating distribution functions (uniform and beta). Each data point in the graph is an average over the corresponding 1000 settings that were originally generated for that combination of N and the generating distribution function. From the figure, we observe a substantial performance improvement with both generating distribution functions (more than 50%, with substantial greater improvement in settings with relatively small N values, i.e., corresponding to real-life settings where the distance between nodes in hops is relatively moderate).

The top of Figure 6 compares $CERS$ with ERS, $BERS$, and our proposed improvement $OBERS$, as a function of N in settings generated using the beta generating distribution function. The left graph depicts the percentage of cases where CERS actually improved performance (rather than resulting with a sequence similar to the one produced by the method we are comparing to). As can be observed from the figure, with comparison to the existing methods, ERS and BERS, our proposed method CERS improves performance in the majority of the cases (converging to 98% for ERS and 100% for BERS). The comparison to our proposed improvement to BERS (the $OBERS$ curve) illustrate how indeed substantial is the use of the indifference point with that method—CERS improves OBERS performance in 70% of the cases, compared to 100% of the BERS solutions, i.e., in about 30% of the cases, the CERS results in an OBERS-like sequence. The middle graph depicts the average overhead reduction achieved when using CERS compared to the three other methods, over all settings where CERS actually improved performance rather than producing a similar sequence. As can be observed from the graph, compared to BERS and OBERS the average performance improvement with CERS decreases as N increases. Compared to ERS, the average improvement actually increases as N increases.

The right graph of Figure 6 depicts the percentage of cases where CERS resulted in a better (rather than equal) performance when taking the best out of ERS and BERS and the best out of ERS and OBERS as a reference. From the graph we observe that compared to the best of ERS and BERS, the CERS nearly always manages to come up

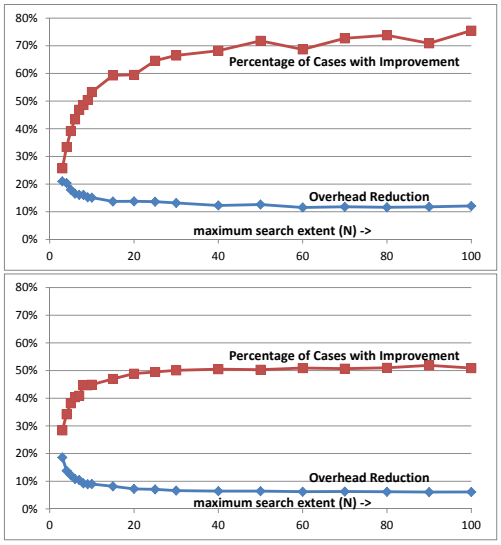

Fig. 8. Adjusted sequence vs. original ERS sequence with no delay constraint - average percentage of overhead reduction and percentage of settings where the sequence was revised, as a function of the maximum search extent N, for: (a) a uniform generating distribution function (top); and (b) a beta generating distribution function (bottom).

with a better sequence. Compared to the best of ERS and OBERS, the percentage of improving cases is also impressive, though substantially smaller. This, once again demonstrates how meaningful the use of our proposed indifference point in BERS is. The reason the two curves in the latter graph increase as N increases is that the greater N is the greater the chance the CERS will find a combination of ERS and BERS steps to be improving. We observe similar qualitative behaviors as those discussed above under the uniform settings at the bottom of Figure 6.

Figure 7 illustrates the methods' performance under time constraints, as a function of the allowed delay (for $N = 5, 10, 20$, represented by graphs in the left, center and right columns, respectively). The top figure focuses on the comparison to ERS and the bottom figure on the comparison to BERS. Both figures use both generating distribution functions (Beta and uniform) depicting both the percentage of cases where CERS results in improved sequence (upper row of graphs) and the average improvement in terms of percentage of reduced overhead in those cases where it improves (bottom row of graphs). Overall, we observe that with the delay constraint CERS manages to improve the solution to a larger portion of the ERS cases, the actual average improvement achieved whenever improving is greater in the BERS-compared cases. The latter observation holds both when using the uniform and the beta generating functions.

Finally, Figure 8 demonstrates the improvement that can be achieved through local adjustments in the optimal ERS sequence as suggested in Section VI. The upper and lower graphs correspond to settings generated using the uniform and Beta generating distribution functions, respectively. The graphs depict both the percentage of cases where a local improvement

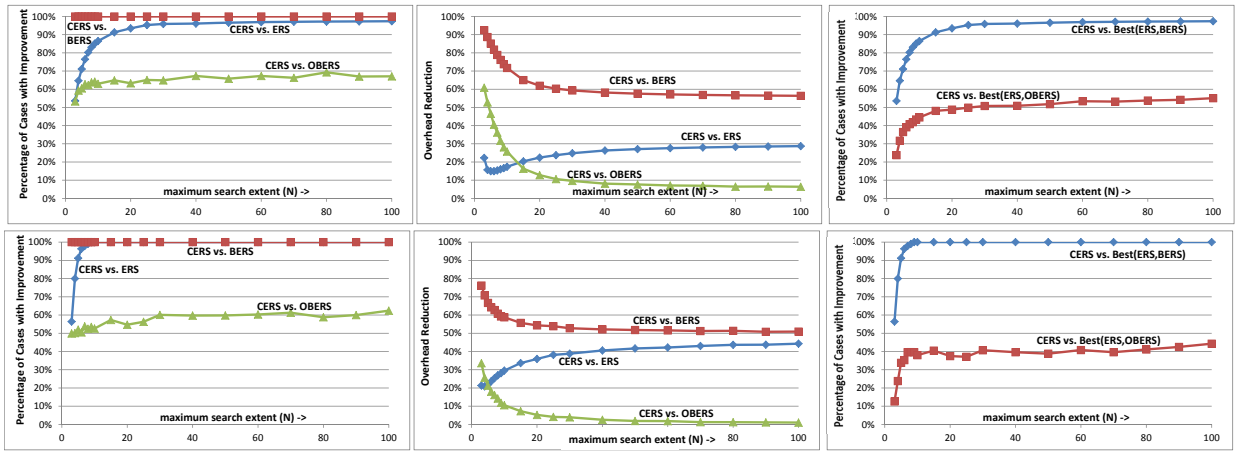

Fig. 6. CERS vs. ERS with no delay constraint as a function of N for the beta generating function (top) and uniform generating function (bottom): percentage of settings where CERS improved performance (left); average percentage of overhead reduction whenever improving (middle); and percentage of settings where CERS improved the best amongst ERS and BERS and amongst ERS and OBERS (right).

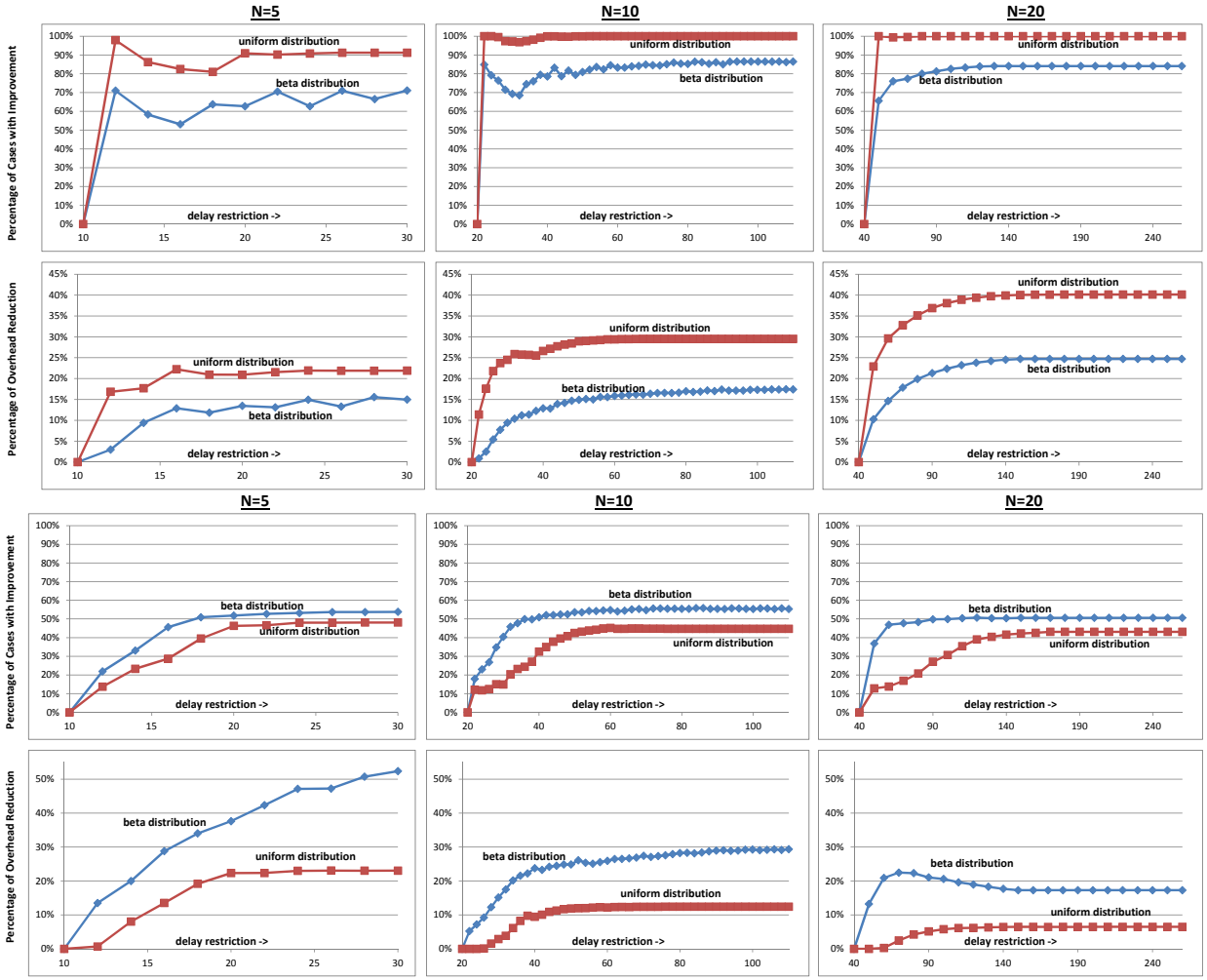

Fig. 7. CERS vs. ERS (top) and vs. BERS (bottom) with delay constraint: percentage of settings where CERS improves performance (upper row) and average percentage of overhead reduction (lower row), whenever improving, as a function of the allowed delay constraint, for different N values (5,10,20) and functions (uniform,beta).

was find applicable and the average improvement achieved in those cases. The increase in the first measure is explained, as before, by having more opportunities to make a change in the sequence as N increases. Similarly the decrease in the second measure is explained by the lower impact each local change has whenever N increases.

IX. Conclusion

We showed how the ERS and BERS techniques can be combined to make a much more efficient search technique–CERS–and how to derive the optimal sequence in quadratic time with respect to the maximum search extent. Since CERS uses the best of either an ERS or BERS type query each round, it is guaranteed to have expected costs at least as low as either of them. We introduced an optimization to BERS that our numerical analysis showed to be very valuable. We showed how to optimize a given sequence in linear time with respect to its length when the probability distribution function changes. This is done by changing the type of query at each extent according to the distribution function. While this does not result in the optimal CERS strategy, it is a quick way to adjust to changing network conditions. Finally, we gave an algorithm to derive the optimal CERS strategy under delay constraints in polynomial time. Our numerical evaluations were highly valuable. They showed that our optimization to BERS alone can reduce overhead by at least 50% or more. Including this optimization in CERS makes it a highly efficient technique, reducing overhead of ERS by up to 40%.

References

[1] J. Ahn and B. Krishnamachari. Scaling laws for data-centric storage and querying in wireless sensor networks. *IEEE/ACM Trans. Netw.*, 17(4):1242–1255, 2009.

[2] M. Al-Rodhaan and A. Al-Dhelaan. Efficient route discovery algorithm for manets. In *NAS*, pages 165–170. IEEE Computer Society, 2010.

[3] H. Barjini, M. Othman, H. Ibrahim, and N. Udzir. Shortcoming, problems and analytical comparison for flooding-based search techniques in unstructured p2p networks. *Peer-to-Peer Networking and Applications*, 5(1):1–13, 2012.

[4] Y. M. Baryshnikov, E. G. C. Jr., P. R. Jelenkovic, P. Momcilovic, and D. Rubenstein. Flood search under the california split rule. *Oper. Res. Lett.*, 32(3):199–206, 2004.

[5] N. B. Chang and M. Liu. Revisiting the ttl-based controlled flooding search: optimality and randomization. In Z. J. Haas, S. R. Das, and R. Jain, editors, *MOBICOM*, pages 85–99. ACM, 2004.

[6] N. B. Chang and M. Liu. Controlled flooding search with delay constraints. In *INFOCOM*. IEEE, 2006.

[7] N. B. Chang and M. Liu. Controlled flooding search in a large network. *IEEE/ACM Trans. Netw.*, 15(2):436–449, 2007.

[8] Z. Cheng and W. B. Heinzelman. Flooding strategy for target discovery in wireless networks. *Wireless Networks*, 11(5):607–618, 2005.

[9] Z. Cheng and W. B. Heinzelman. Searching strategies for target discovery in wireless networks. *Ad Hoc Networks*, 5(4):413–428, 2007.

[10] J. Deng and S. A. Zuyev. On search sets of expanding ring search in wireless networks. *Ad Hoc Networks*, 6(7):1168–1181, 2008.

[11] H. Dubois-Ferrière, M. Grossglauser, and M. Vetterli. Age matters: efficient route discovery in mobile ad hoc networks using encounter ages. In *MobiHoc*, pages 257–266. ACM, 2003.

[12] L. Gargano, M. Hammar, and A. Pagh. Limiting flooding expenses in on-demand source-initiated protocols for mobile wireless networks. In *IPDPS*. IEEE Computer Society, 2004.

[13] J. Hassan and S. Jha. On the optimization trade-offs of expanding ring search. In N. Das, A. Sen, S. K. Das, and B. P. Sinha, editors, *IWDC*, volume 3326 of *Lecture Notes in Computer Science*, pages 489–494. Springer, 2004.

[14] J. Hassan and S. Jha. Optimising expanding ring search for multi-hop wireless networks. volume 2, pages 1061–1065 Vol.2, Nov.-3 Dec. 2004.

[15] S. Jin and H. Jiang. Novel approaches to efficient flooding search in peer-to-peer networks. *Computer Networks*, 51(10):2818–2832, 2007.

[16] Y.-B. Ko and N. H. Vaidya. Location-aided routing (lar) in mobile ad hoc networks. *Wireless Networks*, 6(4):307–321, 2000.

[17] S.-J. Lee, E. M. Belding-Royer, and C. E. Perkins. Scalability study of the ad hoc on-demand distance vector routing protocol. *Int. Journal of Network Management*, 13(2):97–114, 2003.

[18] R. Lima, C. Baquero, and H. Miranda. Broadcast cancellation in search mechanisms. In S. Y. Shin and J. C. Maldonado, editors, *SAC*, pages 548–553. ACM, 2013.

[19] I. Park, J. Kim, and I. Pu. Blocking Expanding Ring Search Algorithm for Efficient Energy Consumption in Mobile Ad Hoc Networks. In *WONS 2006 : Third Annual Conference on Wireless On-demand Network Systems and Services*, pages 191–195, Les Ménuires (France), Jan. 2006. INRIA, INSA Lyon, IFIP, Alcatel. http://citi.insa-lyon.fr/wons2006/index.html.

[20] N. D. Pham, V. V. Zalyubovskiy, and H. Choo. Threshold analysis of expanding ring search for multi-hop wireless networks. In *The 3rd Asia Pacific International Conference on Information Science and Technology (APIC-IST)*, pages 93–102, 2008.

[21] I. M. Pu and Y. Shen. Analytical studies of energy-time efficiency of blocking expanding ring search. *Mathematics in Computer Science*, 3(4):443–456, 2010.

[22] I. M. Pu and Y. Shen. A framework for chase strategies in recent energy or time efficient route discovery protocols for manets. In *WOWMOM*, pages 1–4. IEEE, 2011.

[23] I. M. Pu, Y. Shen, and J. Kim. Measuring energy-time efficiency of protocol performance in mobile ad hoc networks. In D. Coudert, D. Simplot-Ryl, and I. Stojmenovic, editors, *ADHOC-NOW*, volume 5198 of *Lecture Notes in Computer Science*, pages 475–486. Springer, 2008.

[24] I. M. Pu, D. Stamate, and Y. Shen. Improving time-efficiency in blocking expanding ring search for mobile ad hoc networks. *Journal of Discrete Algorithms*, (0):–, 2013.

[25] S. Shamoun and D. Sarne. Two-sided expanding ring search. In *COMSNETS 2014*, pages 1–8. IEEE, 2014.

[26] S. Shamoun, D. Sarne, and S. Goldfeder. Elastic ring search for ad hoc networks. In I. Stojmenovic, Z. Cheng, and S. Guo, editors, *MOBIQUITOUS 2013*, volume 131 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 564–575. Springer, 2013.

[27] S. Tanachaiwiwat, P. Dave, R. Bhindwale, and A. Helmy. Location-centric isolation of misbehavior and trust routing in energy-constrained sensor networks. In *Performance, Computing, and Communications, 2004 IEEE International Conference on*, pages 463–469. IEEE, 2004.

[28] B. Yang and H. Garcia-Molina. Improving search in peer-to-peer networks. In *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*, pages 5–14. IEEE, 2002.

[29] B. Zhang and H. T. Mouftah. Position-aided on demand routing protocol for wireless ad hoc networks. In *ICC*, pages 3764–3768. IEEE, 2004.

[30] H. Zhang and Z.-P. Jiang. On reducing broadcast expenses in ad hoc route discovery. In *ICDCS Workshops*, pages 946–952. IEEE Computer Society, 2005.