Adaptive Broadcast Cancellation Query Mechanism for Unstructured Networks

Rui Lima
INESC Tec, HASLab
Universidade do Minho
Email: rmlima[at]inesctec.pt

Carlos Baquero
INESC Tec, HASLab
Universidade do Minho
Email: cbm[at]di.uminho.pt

Hugo Miranda
LaSIGE - Faculdade de Ciências
Universidade de Lisboa
Email: hmiranda[at]di.fc.ul.pt

***Abstract*—The availability of cheap wireless sensors boosted the emergence of unstructured networks using wireless technologies with decentralised administration. However, a simple task such as learning the temperature needs a discovery service to find a thermometer among all the sensors. In general, resource discovery relies on flooding mechanisms that waste energy and compromises system availability. Energy efficient strategies limit the exploration area, but with a significant impact on latency. The paper proposes ABC (Adaptive Broadcast Cancellation), a new algorithm that uses the knowledge acquired in previous discoveries to accelerate queries towards the resource. Knowledge is stored in a variation of Bloom filters, thus contributing for an efficient utilization of the sensors limited memory.**

Keywords*—*Wireless Networks; Search; Broadcast; Cancellation.

I. Introduction

Many systems have been developed to monitor and locate resources in adverse places. An example are, mining and extraction companies which increasingly use wireless sensors to gather operating conditions data [1]. Low cost wireless sensors can be used to deploy an unstructured network in a mine, as it avoids extending copper wires over several kilometres while permitting to adapt the sampling rate and trigger alarms in a efficient way [2], [3].

Wireless Sensors Networks (WSNs) applications based on periodic reporting communication models [4], [5] are very inefficient in multi-hop networks because they are too resource-intensive (for memory, cpu, power) even when there is no new data to be transmitted. Unstructured sensor networks are developed to maintain communications when topology change, during removing and adding sensor nodes. When trying to locate some target resource, a node queries its neighbours. The simplest approach to implement a query is flooding, where the query is disseminated to all nodes. Unfortunately, successive flooding of the network consumes a lot of resources as it requires a large number of transmissions and occupies bandwidth, contributing to deplete the sensor nodes energy cells [6], [7], [3].

Some energy efficient search mechanisms aim to attenuate the negative impact of flooding by limiting the query scope [8], [9], [10]. This can be achieved by intentionally delaying propagation, exploring a trade-off between energy consumption and response time. This paper proposes a new strategy to improve the performance of flooding protocols, by incorporating a distributed learning mechanism that adapts the protocol behaviour according to previously conducted queries. The paper shows that the proposed **Adaptive Broadcast Cancellation (ABC)** protocol can significantly improve latency and simultaneous reduce the number of retransmissions in long term scenarios, where queries are frequently repeated. ABC uses **Linear Bloom Filter (LBF)**, a Bloom filter inspired mechanism to store, propagate and compact the acquired information in the local vicinity. The ABC protocol was evaluated using the `ns2` network simulation under several network scenarios. Evaluation shows that it is possible to speed up the searching tasks without compromising the energy efficiency.

This work is financed by the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project UID/EEA/50014/2013.

II. Related Work

One of the most simple and popular [8], [11], [12], [13] implementations of broadcast is known as flooding. Flooding is a very energy inefficient [14] mechanism where all nodes retransmit a message when it is received for the first time. One of the reasons for the inefficiency of flooding, comes from the continuation of the message propagation after the resource has been found.

Some solutions have been proposed to stop the flooding, when there is no need to continue the searching process. Limiting the flooding area by using geographically limited broadcasting [15], requires that nodes are aware of their and their peers location, what may not be possible in all scenarios. To control the query searching scope, without global location information, the Expanding Ring Search (ERS) [8] algorithm limits the query scope by adding a Time To Live (TTL) field to the message. Each intermediate node decrements the TTL when it forwards the query message and cease to forward it when TTL reaches 0. ERS works in rounds, if the resource was not found with some initial TTL, then another query is initiated and the TLL value is increased so that the query can go up to the following hops. This process is carried out until the initiator receives a successful answer. ERS will only be energy efficient when the resources are in the firsts hops. Other flooding based mechanisms limit the expanding query using geographical data or distance gradients. Unfortunately a comparison of these methods [15] shows they all exhibit a comparable performance. The Two-Sided Expanding Ring Search (TSERS) [6] considers a searching mechanism where two nodes are simultaneously executing the ERS for a path between each other. All intermediate nodes that receive a query can send back a successful response to the source of the query. TSERS explores the fact that searching cost is lower for shorter distances than for longer ones.

Solutions using chasing packets to stop the query propagation proved to be more energy-time efficient. In the Blocking Expanding Ring Search (BERS*) [16] and in some variations [17], [18], [19] a new control message was added to stop the retransmission process. When some resource is found an acknowledgement is sent back to the initiator node. The initiator sends the stop message, that is forward to every node and stops the expanding query. BERS family protocols are energy efficient when nodes wait a minimum time ($delay \leftarrow f(hop)$) before retransmission. During the delay time window, if a pending retransmission node receives a stop message, the node aborts the forward operation and the expanding query is blocked. However adding the minimum delay suitable for cancellation reduces the time efficiency. There is a trade-off between time efficiency and energy efficiency for protocols using chasing packets. More recent protocols such as Broadcast Cancellation Initiated from Resource (BCIR*) [10] and time-BERS (tBERS) [9], instead of waiting for the initiator node to start the cancellation process, start sending cancellation packets as soon as the resource is found. Both protocols improve time efficiency without additional energy costs.

Linear Bloom Filters (LBF), introduced in this paper, provide both a generalization of Bloom filters [20] and a simple approach to attenuate Bloom filters [21] to capture gradients of resources in a network [7]. The idea of using attenuated Bloom filters and directing routing towards resources is originally in [21]. Their design is based on a vector of standard binary Bloom filters data structures with a probabilistic document localization algorithm; by assigning different weights to each level, a metric similar to our *confidence* metric can be derived. Gradient-based routing using Bloom filters is also used in the Wader design [7]; here the strategy is to decay the information in the filter, by setting random bits to 0 as filters are transmitted, so that resources further away have a lower representation in the local filter. The LBF data structure is a modified Bloom filter capable to store the binary quantification of floating point values. The LBF uses just a single Bloom filter data structure instead of the multiple layered Bloom filters array used by the attenuated Bloom filter technique [21]. The LBF data structure controlled size makes ABC protocol independent of network diameter.

III. An Adaptive Cancellation Mechanism

As discussed above, many broadcast cancellation protocols [16], [10] increase the delay linearly with hop distance. The increasing delay creates a time window to disseminate broadcasting cancellation messages, paving the way to minimize the number of retransmissions, at the expense of time efficiency. None of the protocols BERS* and BCIR* take advantage of the knowledge obtained with previous queries to improve the efficiency of the next. This section describes the **Adaptive Broadcast Cancellation (ABC)** protocol which, by assuming some topology stability, creates a gradient of resource locations that can improve the query dissemination. The ABC protocol abandons the blind delay increase by tuning

Algorithm 1: Linear Bloom Filter (LBF)

```
$lLBF[] \leftarrow [0, \cdots, 0]$;                          // Local LBF
$att \leftarrow 0.9$;                                 // Attenuation factor
$k \leftarrow 4$;                               // Number of hash functions
Function LBFins(elem)
    for $i \leftarrow 1$ to $k$ do
        $idx \leftarrow \mathsf{hash}_i(elem)$;
        $lLBF[idx] \leftarrow 1$;
    end
Function LBFmerge(rLBF[])
    for $i \leftarrow 1$ to $rLBF.size$ do
        $lLBF[i] \leftarrow \max(lLBF[i], rLBF[i] \times att)$;
    end
Function LBFget(elem,lLBF[])
    $c \leftarrow 1$;
    for $i \leftarrow 1$ to $k$ do
        $idx \leftarrow \mathsf{hash}_i(elem)$;
        $c \leftarrow \min(c, lLBF[idx])$;
    end
    return $c$;
```

it with the knowledge gained in previous queries. The key idea is to have nodes propagating queries faster in the direction where the resource is expected to be found. In contrast, when the relay by some node is assumed to be less relevant for a specific query, delay is kept at an adequate level to facilitate the cancellation process.

A. Linear Bloom Filter (LBF)

Knowledge about resource locations is abstracted by each node on a space-efficient probabilistic data structure coined as Linear Bloom Filter (LBF). A traditional Bloom filter is represented by a binary array where each position can store a bit $\{0, 1\}$, that is used to test whether an element is a member of a set or not. With the LBF variation of standard Bloom filters our approach stores floating point values $c \in [0, 1]$ instead of binary values. LBF assign a totality ordered confidence c quantity, to each stored element. This variation extends the traditional test operation, permitting to map each element in a set with a confidence c level, where 0 representing absence of knowledge and 1 the highest confidence. By default c is set to 0 for all LBF positions.

When a node hosts some resources, this information can be stored by setting c to 1 on the positions associated to the resources by the hash functions. In ABC, resources owned by neighbours are added to the same structure by editing the confidence values. Fig. 1 depicts an LBF sample representing the confidence values for elements $\{a, b, c, d, e\}$. By having all their positions with confidence 1, the LBF signals that elements $\{a, e\}$ are stored locally. The high confidence value associated to resource b suggest that it is stored in some node in the neighbourhood, while c and d are hosted by more distant nodes.

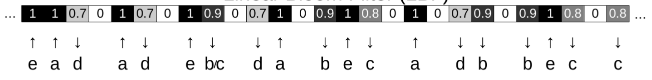


Fig. 1: Node context with LBF

LBF Operations: Alg. 1 defines three functions to operate the LBF data corresponding to the actions: i) inserting elements into a LBF; ii) merging data from two LBF; and iii) getting the

Algorithm 2: Start Query

```
begin
    $pkt \leftarrow creatPacket(\cdots)$;
    $queryList \leftarrow queryList \cup \{msg.queryID\}$;
    $pkt.bloom \leftarrow node.lLBF$;
    $send(pkt)$;
end
```

confidence level for some element. Network resources are designated as $elem_i$ in LBF context. Function $LBFins(elem_i)$ is called to insert some $elem_i$ in node i's LBF. It generates the hash keys to address k independent Bloom array positions, setting them to 1. Function $LBFmerge(rLBF[])$ combines node i's local information ($lLBF[]$) with received neighbour node information ($rLBF[]$). Note that the remote data is attenuated by a constant factor att lower than 1, to privilege local information. Function $LBFget(elem_i, lLBF[])$ returns an estimate of the confidence value c assigned for element $elem_i$. The incorporation of the LBF gossiped in the query dissemination progressively attenuates the confidence values by att. A resource that is expected to be at hop distant, has a confidence level of $c = att^{hop}$.

B. Query Dissemination

To describe the ABC algorithm we assume that a node N_i can have three primitive actions related with each query: N_i can start a new query, by broadcasting a searching message m_s; can start the cancellation process, by sending a cancellation message m_c or; N_i can forward either or both of the messages m_s and m_c. Network communication is asynchronous. Messages m_s and m_c are tagged with a *queryID*, so that N_i can track similar concurrent queries, avoiding message duplication and spurious retransmissions on the WSN.

Before broadcasting the first query there is no network context and each N_i only knows its local resources ($lLBF[]$). However it will be necessary to disseminate the local context through the network, without knowing the number of nodes or topological properties, such as network diameter.

When N_i starts a new query it creates a new packet as illustrated in Alg. 2. A query is initiated with the broadcast of a message m_s. N_i copies $lLBF[]$ into the message header, thus initiating its dissemination. To acquire resource context the $lLBF[]$ is piggybacked in the message and incorporated in the LBFs of the receiving nodes, using function $LBFmerge(rLBF[])$. Learning with a piggybacking approach prevents the dissemination of additional messages. The increase in the size of the payload is not significant and is bounded to the LBF size.

C. Query Handling

As in BCIR⋆ [10], ABC progressively increases query propagation latency to facilitate the dissemination of cancellation messages once the resource is found. This is reflected in the query handling algorithm depicted in Alg. 3. Once a query is received, nodes confirm message validity, discarding it if i) a cancellation for this message was already received or ii) it is a duplicate. Afterwords, the query may produce two outcomes. The node either starts the cancellation process by sending m_c if it hosts the corresponding resource, or schedule the query message m_s for retransmission.

Algorithm 3: Query Handling

```
Function handleQuery (msg)
    $LBFmerge(msg.rLBF)$;
    $pkt.bloom \leftarrow node.lLBF$;
    if $msg.queryID \notin queryList$ then
        $queryList \leftarrow queryList \cup \{msg.queryID\}$;
        if $LBFget(msg.elem, lLBF) = 1.0$ then
            $cancelList \leftarrow cancelList \cup \{msg.queryID\}$;
            $send(CancelPacket(pkt))$;
        else
            $time \leftarrow calcDelay(msg.header, ABC)$;
            $insertTxQueue(time, pkt)$;
        end
    end
Function handleCancellation(msg)
    $LBFmerge(msg.rLBF)$;
    $pkt.bloom \leftarrow node.lLBF$;
    if $IsInTxQueue(pkt)$ then
        $removeTxQueue(pkt)$;
    else
        if $(NodeHostsResource(msg.elem) \wedge$
        $node.resourceFound[msg.queryID] = False$ then
            $node.resourceFound[msg.queryID] \leftarrow True$;
            $msg.resourceHost[msg.queryID] \leftarrow node.ID$;
        end
        if $msg.queryID \notin cancelList \wedge$
        $msg.queryID \in queryList$ then
            $time \leftarrow calcDelay(msg.header, FLOOD)$;
            $insertTxQueue(time, pkt)$;
        end
        $cancelList \leftarrow cancelList \cup \{msg.queryID\}$;
    end
```

Algorithm 4: Added delay

```
const DELAY_MIN ;                                   // Minimum delay
const JIT_MAX ;                                     // Maximum jitter
Function calcDelay (msg,proto)
    $jitter \leftarrow runif(0, \text{JIT\_MAX})$;    // Random Uniform
    $time_{FLOOD} \leftarrow \text{DELAY\_MIN} + jitter$;
    $time_{BCIR^\star} \leftarrow$
    $\max(2 \cdot (msg.hop - 1) \cdot time_{FLOOD}, time_{FLOOD})$;
    $c_1 \leftarrow LBFget(msg.elem, msg.rLBF)$;
    $c_2 \leftarrow LBFget(msg.elem, node.lLBF)$;
    if $c_1 < c_2$ then
        $time_{ABC} \leftarrow \max(2 \cdot (msg.hop - 1) \cdot \text{DELAY\_MIN} \cdot$
        $(1 - c_2) + jitter, time_{FLOOD})$;
    else
        $time_{ABC} \leftarrow time_{BCIR^\star}$;
    end
    return $now() + time_{proto}$;
```

The ABC added delay applied during the query dissemination follows Alg. 4 and considers 3 components. The resource location independent component is given by a static value (DELAY_MIN) plus some $jitter$, modelled by a random uniform distribution. The location dependent component makes the delay proportional to the message hop counter (see Alg. 3). This is the component that facilitates message cancellation. The third component is dictated by the nodes estimation of its distance to the resource, given by the LBF prediction. This component given by $(1 - c_2)$ (see Alg. 4) aims to speed up the query propagation in the predicted direction of the node hosting the resource. The $calcDelay(msg, proto)$ function in Alg. 4 returns the schedule time for message retransmission.

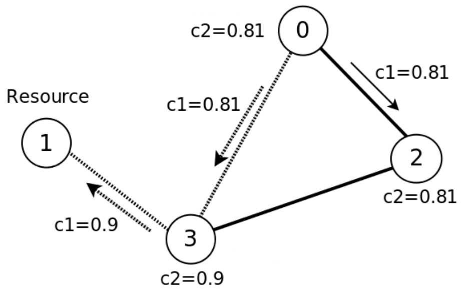


Fig. 2: ABC - Delay Attenuation

Fig. 2 depicts an example of a query propagation accelerated into the most likely direction. It considers that N_0 starts looking for a resource located at N_1 and assumes that nodes know the confidence c associated with the expected hop distance. N_3 receives m_s and given that $c_1 < c_2$, an attenuation $(1 - c_2)$ is applied to the delay, speeding up the query towards node N_1. On the other hand, given that N_2 receives m_s and $c_1 \geq c_2$, no attenuation will be applied.

IV. Evaluation

Ad-hoc network topologies can be modelled using Random Geometric Graphs (RGG) or approached by synthetic topologies, such as those of the Manhattan movement model. This paper evaluates both of these topology classes, while examining some more extreme deployments, occurring in mines [1], that exhibit higher path lengths. To evaluate the performance of ABC and understand the impact of the adaptive delay on the number of transmissions and end-to-end delay, an implementation for the ns2 network simulation was developed. ABC is compared with a naive implementation using plain flooding and BCIR*, that one of the most efficient protocols to control the query searching scope. ABC includes the adaptive delay while other broadcast cancellation protocols operate without knowledge from previous queries.

Two metrics are used to study the ABC time and energy efficiency behaviour, namely end-to-end delay and the number of transmissions. The **end-to-end delay (L)** is defined as the average time between the broadcasting search by the initiator node N_0 and the moment at which N_0 receives the first answer. The energy cost is measured by the average **number of transmissions (R)** performed by every node on each query. It is assumed that any two retransmissions consume the same amount of energy.

A. Network Topologies

Two topology models are used to create Manhattan scenarios for ns2. The Manhattan scenarios are set up with equidistant streets from each other, ensuring that the interior streets share equitably distance to cover the area. The distance between the streets is greater than the radio range, to avoid direct communication from one street to another. The **equidistant grid** was generated with 176 nodes uniformly distributed along the rows covering an area of $3200 \times 3200\text{m}^2$, resulting in a node density of $\simeq 17$ Nodes/Km^2. The **Manhattan** topology was generated using the BonnMotion [1] tool with 250 nodes distributed along the streets covering an area of $2000 \times 2500\text{m}^2$, resulting in a node density of 50 Nodes/Km^2.

[1]http://sys.cs.uos.de/bonnmotion/index.shtml

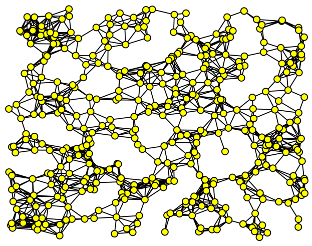


Fig. 3: Random Geometric Graph (N≃400)

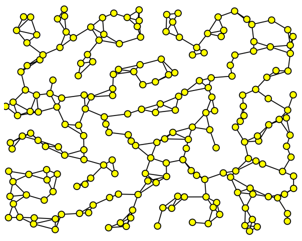


Fig. 4: *Modified* Random Geometric Graph (N≃165)

TABLE I: ns2 Simulation Parameters

Attribute	Value [Default]
DELAY_MIN	10 ms
jitter	7 ms
Packet Size	1000 bytes
Max Simulation Time	17000s
Number of topologies	500
Radio range	250 m
Network links	11 MBit/s
Transmission power	25 dBm
Antenna gain	1
Antenna Polarization	Omnidirectional
ns2 propagation model	TwoRayGround

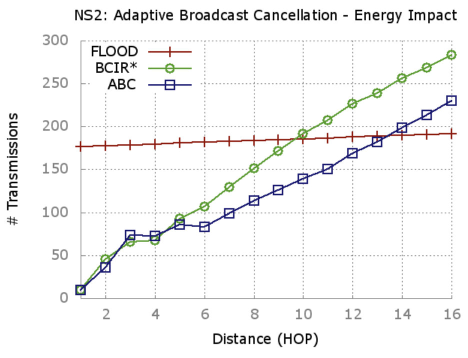
The creation of the **mine topologies** mirrors previous works [22] [23] using random geometric graphs (RGG) to model wireless networks topologies. The bi-dimensional random geometric graph model $G(N, r, \ell)$ creates N nodes that are randomly placed inside an unit square area. The topology defined in the unit square is mapped into the ns2 using a linear transformation taking into account the radio transmission range. Any two nodes whose euclidean distance is below r are considered connected and linked by an edge. Unfortunately the RGGs created using NetworkX[2] reveal a graph topology that was not suitable to represent mine topologies scenarios because there were too many clusters with high level of redundancy. To create more realistic scenarios, a maximum graph degree was defined by removing nodes whose degree exceeds a threshold. The result is a *modified* random geometric graph. Figure 3 depicts the original RGG as created by NetworkX and Figure 4 depicts the modified version of the same graph, after imposing a limit to the node degree. The mine topologies use $N \simeq 165$ nodes on a square area of $3100 \times 3100\text{m}^2$.

The modified RGG is considered to make a better mine structure representation, resulting in a topology with longer paths (tunnels) and lower node density.

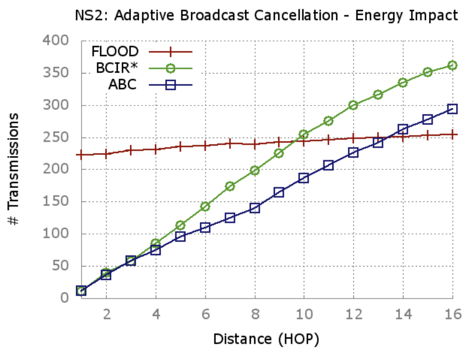
B. Experiment Setup

Table I summarizes the parameters used for setting ns2 simulations. The antenna settings, signal power and receiving threshold, are selected to give approximately 250m of radio communication range. To each node is attached one resource. The queries are schedule in random order and search for random resources.

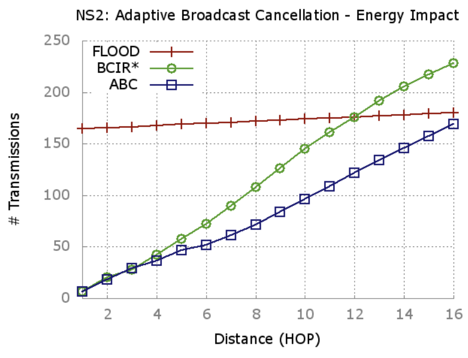
[2]http://networkx.github.io



(a) Equidistant Grid Topology



(b) Manhattan Topology



(c) Mine Topology

Fig. 5: Impact of the topologies and distance on the number of transmissions.

The ABC evaluation tests use a default LBF with $m = 256$ cells and $k = 4$ hash functions. These constants are suggested by Eq. (1), extracted from [7], to provide a false positive probability up to $fp = 5\%$ and considering that LBF can keep information for at least 40 resources (i.e. $\#LBF < 40$).

$$m = -\frac{\#LBF \cdot \ln(fp)}{(\ln 2)^2} \qquad ; \qquad k = \ln(2) \cdot \frac{m}{\#LBF} \qquad (1)$$

The attenuation factor att (used in Algorithm 1) can be any value from $[0, 1]$ and is set to $att = 0,9$ corresponding to 10% of attenuation for each hop.

C. Result Analysis

Results are presented as a function of the distance between the initiator and the first node responding with a successful answer, a metric hereafter designated as hop count.

All protocols are submitted to a pool of 500 topologies. For each topology, the queries are scheduled using a uniform distribution over the maximum simulation time. The absolute number of queries that received a successful reply is depicted in Fig. 7. Table II lists the percentage of packet drops for each protocol, which confirms density and topology impact on the algorithms performance. Not surprisingly, for the equidistant grid and Manhattan topologies, the number of packets dropped increases with node density, an aspect that can be attributed to the increasing number of collisions.

TABLE II: Packet Drops

TOPOLOGY	Nodes/Km^2	FLOOD	BCIR*	ABC
Mine	$\simeq 17$	$\simeq 5\%$	$\simeq 3\%$	$\simeq 3\%$
Equidistant Grid	$\simeq 17$	$\simeq 11\%$	$\simeq 10\%$	$\simeq 10\%$
Manhattan	$\simeq 50$	$\simeq 30\%$	$\simeq 32\%$	$\simeq 33\%$

The impact of packet drops can be observed in Fig. 7, which shows the absolute number of successfully replied queries in the mine scenario. A comparison with table II suggests a correlation between the number of successfully queries and the number of packets dropped.

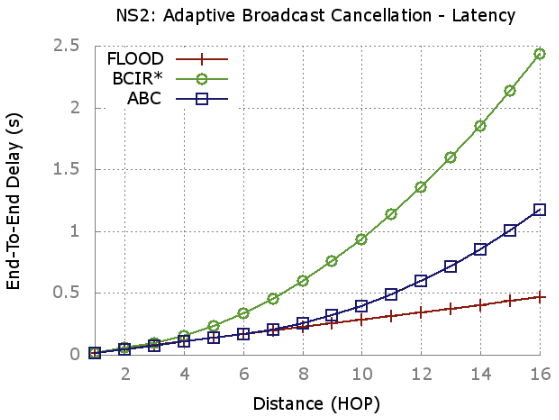


Fig. 6: End-To-End Delay

Energy impact. Assuming that transmissions are the dominant factor on energy consumption in a system based on wireless communications, we compare the energy efficiency by counting the number of transmissions used until all messages related with a query cease to be propagated. As depicted in Fig. 5, the energy consumption of ABC is smaller than BCIR* for queries to resources at 6 or more hops. A comparison between the mine and the equidistant grid topologies shows that the adaptation mechanism of ABC preserves the energy consumption in a pattern comparable to BCIR*, specially in the first hops. This is very significant considering that ABC was able to improve latency and that BCIR* is capable of significantly reducing transmissions with respect to flooding.

Latency. Fig. 6 shows the variation of the average end-to-end delay between the moment a query is broadcast and its sender receives a reply. The ABC protocol significantly reduces the average end-to-end time and the gains increase with the distance to the resource (Fig. 6). For example, the difference between BCIR* and the FLOOD baseline for $H = 4$ is similar to the difference between ABC and the FLOOD baseline for $H = 8$. This result can be attributed to the LBF contribution for accelerating the query propagation in the resource direction, a unique property of ABC which makes it competitive with flooding for resources located up to 7 hops.

Both ABC and BCIR* protocols evidenced a trade-off between latency and the number of transmissions, and by

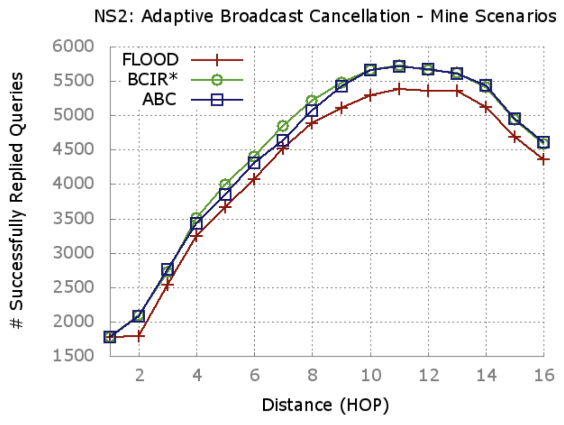


Fig. 7: Successfully Replied Queries (Mine Scenarios)

comparing the performance for resources at the same distance, ABC significant gains in time with faster responses and with a reduction in the number of transmissions. The gains increase in line with the distance between the source of query and the resource. However, a cross-analysis of Fig. 5 and Fig. 6 indicate that ABC is able to reduce the latency without increasing transmissions cost.

Fig. 7 shows that the number of successfully replied queries for ABC is similar to BCIR* and by combining with the latency information from Fig. 6, the results show a very significant reduction in latency, justifying the effort to reduce the added delay achieved by ABC.

V. Conclusions

Mechanisms to improve the performance of resource discovery in WSNs typically exhibit a trade-off between the number of transmissions and latency. This paper presented ABC, an algorithm that makes an innovative approach capable to attenuate such trade-off by speeding the query propagation in the predicted direction of the resource. Evaluation showed that a sense of the resource location can significantly reduce the end-to-end time, while still benefiting from a reduction in energy cost for successive discovery queries. ABC achieves these goals with a small penalty in the query message size, resulting from the piggyback LBFs inside query messages. The ABC protocol operates by taking advantage of the query propagation mechanism, to epidemically disseminates the resource location direction without increasing the number of transmitted messages.

References

[1] M. Li and Y. Liu, "Underground coal mine monitoring with wireless sensor networks," *ACM Trans. Sen. Netw.*, vol. 5, no. 2, pp. 10:1–10:29, Apr. 2009. [Online]. Available: http://doi.acm.org/10.1145/1498915.1498916

[2] G.-R. Lin, Y.-C. Fan, E. T. Wang, T. Zou, and A. Chen, "Energy-efficient sensor data acquisition based on periodic patterns," in *Parallel and Distributed Systems (ICPADS), 2009 15th International Conference on*, Dec 2009, pp. 487–494.

[3] M. Maróti, "Directed flood-routing framework for wireless sensor networks," in *Proceedings of the 5th ACM/IFIP/USENIX International Conference on Middleware*, ser. Middleware '04. Springer-Verlag New York, Inc., 2004, pp. 99–114.

[4] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh, "Efficient network flooding and time synchronization with glossy," in *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, April 2011, pp. 73–84.

[5] L. Mottola and G. Picco, "Muster: Adaptive energy-aware multisink routing in wireless sensor networks," *Mobile Computing, IEEE Transactions on*, vol. 10, no. 12, pp. 1694–1709, Dec 2011.

[6] S. Shamoun and D. Sarne, "Two-sided expanding ring search," in *Communication Systems and Networks (COMSNETS), 2014 Sixth International Conference on*, Jan 2014, pp. 1–8.

[7] D. Guo, Y. He, and Y. Liu, "On the feasibility of gradient-based data-centric routing using bloom filters," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 25, no. 1, pp. 180–190, Jan 2014.

[8] D. B. Johnson, D. A. Maltz, and J. Broch, "Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks," in *In Ad Hoc Networking, edited by Charles E. Perkins, Chapter 5*. Addison-Wesley, 2001, pp. 139–172.

[9] I. M. Pu, D. Stamate, and Y. Shen, "Improving time-efficiency in blocking expanding ring search for mobile ad hoc networks," *J. of Discrete Algorithms*, vol. 24, pp. 59–67, Jan. 2014. [Online]. Available: http://dx.doi.org/10.1016/j.jda.2013.03.006

[10] R. Lima, C. Baquero, and H. Miranda, "Broadcast cancellation in search mechanisms," in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, ser. SAC '13. ACM, 2013, pp. 548–553. [Online]. Available: http://doi.acm.org/10.1145/2480362.2480467

[11] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Mobile Computing Systems and Applications, 1999. Proceedings. WMCSA '99. Second IEEE Workshop on*, Feb. 1999, pp. 90–100.

[12] N. Beijar, "Zone routing protocol (zrp)."

[13] Y.-B. Ko and N. H. Vaidya, "Location-aided routing (lar) in mobile ad hoc networks," *Wirel. Netw.*, vol. 6, no. 4, pp. 307–321, Jul. 2000. [Online]. Available: http://dx.doi.org/10.1023/A:1019106118419

[14] S. Preethi and B. Ramachandran, "Energy efficient routing protocols for mobile adhoc networks," in *Emerging Trends in Networks and Computer Communications (ETNCC), 2011 International Conference on*, april 2011, pp. 136 –141.

[15] Q. Chen, S. S. Kanhere, and M. Hassan, "Performance analysis of geography-limited broadcasting in multihop wireless networks," *Wireless Communications and Mobile Computing*, vol. 13, no. 15, pp. 1406–1421, 2013. [Online]. Available: http://dx.doi.org/10.1002/wcm.1188

[16] 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*. INRIA, INSA Lyon, IFIP, Alcatel, Jan. 2006, pp. 191–195, http://citi.insa-lyon.fr/wons2006/index.html.

[17] I. Pu and Y. Shen, "Enhanced blocking expanding ring search in mobile ad hoc networks," in *New Technologies, Mobility and Security (NTMS), 2009 3rd International Conference on*, dec. 2009, pp. 1 –5.

[18] M. A. Al-Rodhaan, L. Mackenzie, and M. Ould-Khaoua, "Improvement to blocking expanding ring search for manets."

[19] A. Shintre and S. Sondur, "Improved blocking expanding ring search (i-bers) protocol for energy efficient routing in manet," in *Recent Advances and Innovations in Engineering (ICRAIE), 2014*, May 2014, pp. 1–6.

[20] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Commun. ACM*, vol. 13, no. 7, pp. 422–426, Jul. 1970. [Online]. Available: http://doi.acm.org/10.1145/362686.362692

[21] S. Rhea and J. Kubiatowicz, "Probabilistic location and routing," in *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, 2002, pp. 1248–1257 vol.3.

[22] H. Kenniche and V. Ravelomananana, "Random geometric graphs as model of wireless sensor networks," in *Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on*, vol. 4, feb. 2010, pp. 103 –107.

[23] M. Haenggi, J. Andrews, F. Baccelli, O. Dousse, and M. Franceschetti, "Stochastic geometry and random graphs for the analysis and design of wireless networks," *Selected Areas in Communications, IEEE Journal on*, vol. 27, no. 7, pp. 1029–1046, September 2009.