MuTrans: A Multi-Channel Network Coding Approach to Mobile Data Collection

Mansour Abdulaziz
Department of Computer Science
George Mason University
Fairfax, Virginia 22030
mabdulaz@gmu.edu

Robert Simon
Department of Computer Science
George Mason University
Fairfax, Virginia 22030
simon@gmu.edu

***Abstract*—The advantages of using of the mobile sinks (MSs) to perform data collection from Wireless Sensor Networks (WSNs) are now widely recognized. This is because the MS data collectors can service isolated systems and reduce energy expenditures by minimizing the need for multi-hop networking. However, it remains a challenging problem to support collection activities efficiently within the relevant class of predictable but uncontrollable MSs. This paper presents *MuTrans*, a novel multi-channel protocol that uses both clustering and network coding techniques to increase the reliability and reduce the latency of data collection in predictable and uncontrollable MS systems. A fairness data uploading scheduling to a mobile collector is presented based on assigning a method for load balancing. The implementation is described by using multiple channels for increased throughput and incorporating network coding for improved reliability. Our evaluation of data aggregation in the presence of packet errors shows that MuTrans can significantly reduce the latency for data collection, thus providing strong support for mobile data collection.**

I. Introduction

Low-power and Lossy Networks (LLNs) are embedded systems with limited resources such as processing, power, and memory [8]. They are lossy because they use wireless communication that has a high error rate and unpredictable reception. Due to these constraints, data collection in dense and highly distributed WSNs that require long multi-hop routing paths to data sinks can significantly increase management complexity, expend excessive amounts of power in relaying packets or, in the case of battery-powered WSNs, degrade network lifetime [2]. On the other hand, using mobile data gathering can save sensors battery life and diminish the number of relay-only nodes that need to be deployed. Mobile data collection is also suitable to retrieve data from networks deployed in isolated and hard to reach locations.

Despite the known advantages of mobile data collection for many types of LLNs, it remains a challenging problem to effectively support mobile data collection. This is due to the fact that LLN nodes generally suffer from high bit and packet error rates. To address this problem, a wide range of mobile data collection application classes has been broadly investigated [3].

In this paper, we focus on uncontrollable predictable mobile applications. This type of MS traverses the network in a predetermined path with an arrival schedule that is known in advance, without motion control. An example of such an application: a public transportation vehicle (e.g. bus or train) in Smart Cities [7]. While the vehicle is in service, it collects data from surrounding nodes and delivers it to the appropriate base station (BS).

Reducing the latency of data collection becomes an issue. This is because the MS is only in the range of each node for a fixed amount of time, so if some data is not collected it must wait until the next time the mobile comes in ranges. Our paper addresses this problem by designing and evaluating a novel multi-channel LLN protocol that also takes advantage of network coding for uncontrollable predictable mobile data collection. We call our protocol *MuTrans* for *Mu*lti-channel *Trans*port.

By transmitting on different frequencies, multi-channel LLN networking can both reduce delivery latency and improve system throughput [5]. For wireless networks that have high bit error rates, network coding is known to significantly improve throughput [4].

II. Background and Related Works

Our approach assumes the ability to form one hop clusters in a controllable predictable mobile data collection environment. Example work in this area includes Load Balanced Clustering with Dual Data Uploading (*LBC-DDU*) [9]. LBC-DDU constructs energy-balanced clusters while having a mobile data collector to aggregate data from cluster heads in a reliable communication using two antennas. LBC-DDU carefully picks what so called *polling points* from a set of candidate polling points. The polling points are the locations the mobile collector has to stop by and retrieve the data to send them later to the base station. LBC-DDU converts any form of routing into a one-hop path by creating balanced single-hop clusters.

By considering the changes in mobile sink trajectories, Smeets et al. [7] implemented a mobile platform called *Trainsense* to support mobile WSNs applications. The modeled train inheres some of its features – such as controllers and detectors – to the motes for different mobility trajectories. Nevertheless, none of these related works have considered fixed route and uncontrollable mobile data collectors with one-hop routing in unstable environments.

By considering this mobility model, we propose two heuristic methods to reduce the uploading latency by ap-

ISBN 978-3-901882-88-3 © 2017 IFIP

plying the advantage of network coding and multi-channel coding protocol in LLN networks [1]. This protocol (called *MuCode*) allows head nodes the possibility to overhear packets synchronously from their member nodes, while also eavesdropping on packets from other nodes in order to increase the chances of recovering lost data. Each head node concurrently sends all its data to the sink through different channels in order to avoid collision with one condition, that the mobile sink has enough multi-radios to receive packets simultaneously.

III. System Architecture

We assume the route of the mobile collector (MuCar) is known, and the head nodes are determined using the LBC-DDU algorithm, an algorithm that is described in Section 2. We now precisely define *polling sector* as the range in which the MuCar can communicate with head nodes and receive data while it is on the move. The polling sector length at head node H_i can be measured by the distance between the starting point, where MuCar contacts with H_i, and the ending point before it disconnects with H_i. The location of each polling sector is known once the head nodes are selected.

Members transmit their data to the nearby head nodes where the data is stored. Each member node picks one head node to send their data to. We are assuming a non-splitting data flow system, which means that the data from a node has to be sent to its dedicated neighbored node without dividing the data into multiple receivers. Once MS initiates a contact with the head nodes, they send their aggregated data along with their own data to MS.

LLN nodes possess radios that have single transceivers and are capable of supporting multi-channel communication protocols [6]. The sink, on the other hand, has multiple radios that allow it to receive data from head nodes concurrently through different channels. This multiple radio setup is used to prevent packet collisions and reduce uploading latency. By enabling multi-channel networking, head nodes in a neighborhood can send packets simultaneously without causing collisions, as long as they use different channels (λ_i). The gap between any two frequencies is large enough so that the interference is eliminated, meaning that channel λ_i does not interfere with λ_j, where $i \neq j$. Successful use of multi-channel communication requires that senders and receivers agree upon which channels to use at which time, and their agreement requires a channel scheduling policy. The advantage of this approach is that any pair of nodes that use different channels will not interfere with each other, thus avoiding the hidden terminal problem.

Figure 1 illustrates the MuTrans system architecture. The basic idea is that there are two types of nodes – head nodes and member nodes. The Figure shows the member node m_3 sending its packets to its head node (solid line) H_3 , while another head node H_4 overhears the packets (dotted line) and uses network coding to enhance network reliability.nOnce the mobile sink (MS) enters the polling sector, the head nodes transmit their stored data through different assigned channels λ_i.

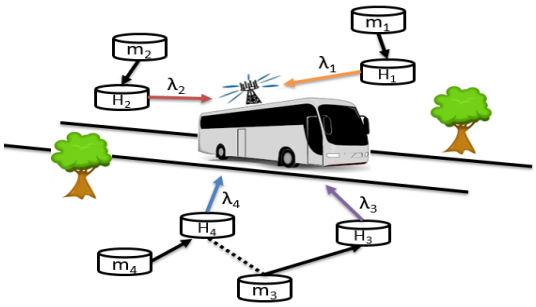

Fig. 1: MuTrans System Architecture

The member node assignment to each head node becomes critical in this situation. This assignment can increase the data load at the head node, which increases the energy consumption while also increasing the uploading latency to the MS. Since the MS has uncontrollable mobility, the uploading deadline is firm and unbreakable at each polling sector. Another important consideration is when the number of concurrent uploading head nodes exceeds the number of radios available at the MS. Therefore, fair and balanced scheduling mechanisms have to be designed to overcome this problem.

IV. MuTrans Protocol

The purpose of the $MuTrans$ protocol is twofold: First, to dynamically assign each member node to a head node. Second, to, as fairly as possible (in the sense of overall reducing total upload latency), assign uploading schedules to head nodes. For each of these problems, we present polynomial time heuristics. The heuristics are meant to be run with full knowledge of system parameters, and therefore can be run by the MS. The protocol can be run continuously as the mobile traverses the system or can be run whenever system parameters change, such as head node or member node assignment, changes in data rates, or changes in mobility patterns as in LBC-DDU.

A. Data Load Balancing

The Data Load Balance (*DLB*) algorithm heuristically balances data loads among the head nodes in order to reduce the uploading duration in the mobile data collector. We define $C(m_i)$ as the set of candidate head nodes that the member node m_i can assign its data to. Those candidates are one-hop distanced to m_i. Initially, each member and head node calculates its collected data $\Psi_{m_i}(\sigma)$ and $\Psi_{H_i}(\sigma)$ respectively.

The set Q is defined as having all of the member nodes at the current polling sector so that it is sorted based on the number of candidate head nodes per member nodes in ascending order and then based on the local data in descending order. In this way, the least candidates' member node chooses first in order to give more precise and better options to the member nodes that have more candidate head nodes. If we do the opposite, then a member node with high candidates may choose a head node, which may be the only option for another member node – therefore increasing the

load to that head node. This can be avoided if that choice was processed later.

Then, iteratively, we pick the first member node and choose the head node with lowest data load and update its data load by including the member node's data load. This iteration is processed until all the elements in the set Q are selected. The complexity time for our heuristic algorithm is $\mathcal{O}(\mu_j$ x $(\Gamma_j + \log \mu_j))$, where μ_j is the number of member nodes, while Γ_j is the number of head nodes.

B. Utilized Fair Scheduling

A synchronized schedule that is based on the number of packets in the head nodes has to be maintained to reduce the uploading latency. Given Γ_j, and given head nodes that want to upload their data to the mobile data collector at a given polling sector PS_j, we define the Dynamic Round-Robin Scheduling algorithm (*DRRS*) , which fairly and dynamically utilizes the scheduling of the concurrent head nodes that are uploading. Let HS_j be the head node set at the polling sector PS_j.

The iteration of the algorithm considers the remaining non-sent packets in the set HS_j. At first, the set is sorted based on the number of packets each head node has in descending order. Then, it divides these head nodes by the number of available channels that the mobile data collector has, which creates g schedules of head nodes. Each schedule x has a head node with a minimum number of packets (called G^x_{min}), which is the indicator for how many simultaneous transmitting should be done during this iteration. After the transmitting process is complete in this iteration, each head node at schedule x deducts G^x_{min} from its total packet, then HS_j removes all the head nodes that have no remaining packets to send. The algorithm repeats the previous steps until the head nodes transmit all packets. The running time complexity of this algorithm is $\mathcal{O}(\Gamma_j^2 \log \Gamma_j)$.

V. Evaluation and Results

Using Cooja, an LLN simulation tool, we evaluated $MuTrans$ against several other approaches. The purpose of this evaluation was to judge the effectiveness of our proposed heuristic. Since there do not exist other heuristic algorithms that are applicable in our predictable multi-channel environment, we defined two simple heuristics to solve our two subproblems: Random Member Assignment RMA and Static Round Robin Scheduling $SRRS$. RMA is an algorithm that randomly assigns member nodes to their nearby head nodes without considering data balancing, whereas $SRRS$ is a round robin scheduling that fixes the head node location to its original schedule.

These two baseline heuristics enabled us to define three new protocols. The first combination is RMA + $SRRS$, where the head node selection is randomly assigned by the member nodes, while the concurrent uploading schedules are statically assigned to the head nodes with a single iteration. The second combination is RMA + $DRRS$, where the head nodes are selected randomly by the member nodes, while the concurrent uploading schedules are dynamically assigned to the head nodes with multiple iterations as in *DRRS* algorithm. The last combination is DLB + $SRRS$. Here the head nodes are assigned by the member nodes based on *DLB* algorithm, which balances the data rate load at the head nodes; however, the concurrent uploads are statically scheduled to these head nodes. All of these three new protocols use multi-channel network coding.

To represent a realistic uncontrollable but predictable environment, we used a circular topological representation of a University. We used the Contiki operating system to implement the protocols. Cooja, the Contiki simulation tool, emulates the network nodes and its hardware platform. We chose the TMote Sky node which transmits 250 kbit/s using MSP430 microprocessor and CC2420 radio. The platform works in 10 KB of random access memory and 48 KB of program flash. The radio medium is configured to the Unit Disk Graph Medium Distance Loss with 50m transmission and interference ranges. In our experiments, we assumed that each node sends a packet of size 50 bytes at time slot t where the difference between two consecutive time slots is 100 milliseconds per transmission.

The MS is traversing the University in a fixed route with a distance of 5.3 km and a speed of 55 km/h. We installed 16 polling sectors, where the number of head nodes is 6 per polling sector. The total number of deployed nodes are 192 with Δ = 4.6 seconds contact duration between the MS and the head nodes. Each member node i has its own data transmitting rate γ_i in order to test the data balance algorithm. These member nodes are randomly scattered and are 1-hopped away from the head nodes. We evaluated our protocols with a different number of radios $\Re$ available at the MS. These experiments are conducted with a variety of wireless Packet Error Rates (PERs) using the *MuCode* network coding protocol [1].

A. Network Throughput

Figure 2 shows the average network throughput with different PERs. The four protocols shown with $\Re = 2$. At PER = 0%, the throughput in $DLB + DRRS$ outperforms the other method with $\Re = 2$ from 3.7% to 12%. The reason for this superiority is that $DRRS$ allows more packets to be transmitted; however, RMA creates unbalanced data loading among the head nodes, which affects the uploading latency. Meanwhile, DLB effectively balances the data load among the head nodes that reduce the uploading latency, which eventually increases the network throughput with a fixed contact time.

On the other hand, $DLB + DRRS$ $(\Re = 6)$ outperforms the same model when $\Re = 2$ and $\Re = 4$ in relation to network throughput in different packet error rates up to 166% and 73%, while at PER = 40%, the gap is shrunk slightly to 154% and 72%. This throughput's advantage is because $DRRS$ increases the concurrent data uploading by increasing the number of radios available in MS in order to be equal to the number of head nodes at that polling sector, which increases the total number of packets efficiently.

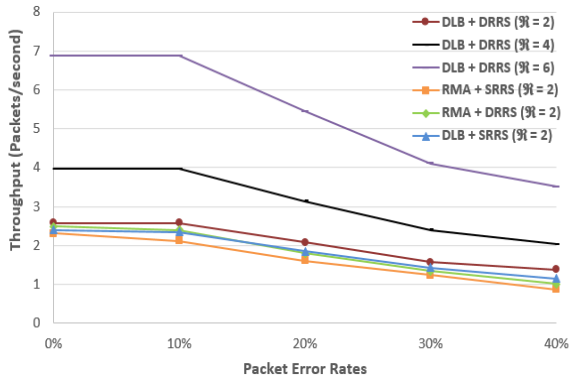

Fig. 2: Average network throughput versus PER with different $\Re$.

B. Trips Required for Data Delivery

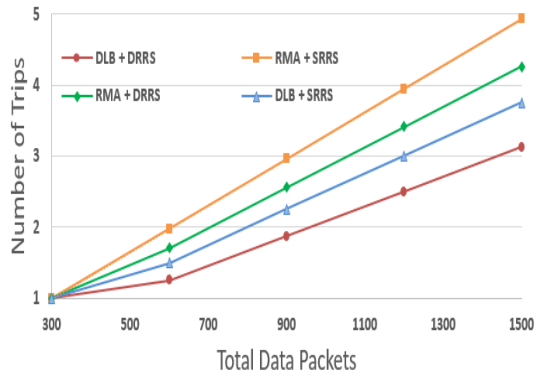

Fig. 3: The average number of trips MS has to take to collects all the data with PER = 40% and $\Re = 2$.

One of our motivations is to quantify the number of trips required to complete data delivery. To assess this, we ignore the impact of buffer overflow and simply count the number of required trips. Figure 3 depicts the average number of trips that MS ($\Re = 2$) has to accomplish in order to get the data needed by the system that has 40% of PER, with different total data packets originated by all the nodes on the network.

With a small amount of data, even with the high packet error rate, all of the four protocols can get all of the data that is originated by the nodes. There are two reasons for this successful transmission: First, network coding is implemented in all of them, which helps recover lost data packets. The second reason for the completed transmission is that the contact duration meets the deadline for uploading the needed data packets without the need for retransmissions.

Once the data packets increase to 900, the MS has to take another trip in order to get the remaining lost packets that needed to be retransmitted. Since the contact duration time is always fixed in such an application, the over amount data packets, which are not considered by the system administrator, will miss the deadline. Hence not only are the retransmission of packets needed in high packet loss rates, but also the MS has to take more trips in order to receive all of the data packets, as seen in the previous figure. Nevertheless, $DLB + DRRS$ outperforms $RMA + SRRS$, $RMA + DRRS$, and $DLB + SRRS$ by up to 57%, 36%, and 20% in data upload latency respectively.

VI. Conclusion

This paper presented *MuTrans*, a network-coded multi-channel protocol for uncontrollable but predictable mobile data transport. Network coding has been used to significantly enhance the reliability and throughput for low power and lossy networks. Multi-channel networking, on the other hand, can essentially reduce the uploading latency and improve the network throughput. *MuTrans* balances the non-reachable node assignments to the local head nodes. It uses synchronized dynamic round robin scheduling for uploading data to the mobile data collector. We evaluate two heuristic algorithms against different heuristic techniques. The results indicated that *MuTrans* outperforms the other methods in packet throughput, and latency.

Acknowledgment

This work is supported by NSF under grants CNS-1116122 and CNS-1205453.

References

[1] M. Abdulaziz and R. Simon. Multi-channel network coding in tree-based wireless sensor networks. In *Computing, Networking and Communications (ICNC), 2015 International Conference on*, pages 924–930. IEEE, 2015.

[2] M. Di Francesco, S. K. Das, and G. Anastasi. Data collection in wireless sensor networks with mobile elements: A survey. *ACM Trans. Sen. Netw.*, 8(1):7:1–7:31, Aug. 2011.

[3] Y. Gu, F. Ren, Y. Ji, and J. Li. The evolution of sink mobility management in wireless sensor networks: A survey. 2015.

[4] L. Keller, E. Atsan, K. Argyraki, and C. Fragouli. Sensecode: Network coding for reliable sensor networks. *ACM Trans. Sen. Netw.*, 9(2):25:1–25:20, Apr. 2013.

[5] Y. Kim, H. Shin, and H. Cha. Y-mac: An energy-efficient multi-channel mac protocol for dense wireless sensor networks. In *Proceedings of the 7th international conference on Information processing in sensor networks*, pages 53–63. IEEE Computer Society, 2008.

[6] M. Sha, G. Hackmann, and C. Lu. Real-world empirical studies on multi-channel reliability and spectrum usage for home-area sensor networks. *Network and Service Management, IEEE Transactions on*, 10(1):56–69, 2013.

[7] H. Smeets, C.-Y. Shih, M. Zuniga, T. Hagemeier, and P. J. Marrón. Trainsense: a novel infrastructure to support mobility in wireless sensor networks. In *Wireless Sensor Networks*, pages 18–33. Springer, 2013.

[8] T. Watteyne, A. Molinaro, M. G. Richichi, and M. Dohler. From manet to ietf roll standardization: A paradigm shift in wsn routing protocols. *Communications Surveys & Tutorials, IEEE*, 13(4):688–707, 2011.

[9] M. Zhao, Y. Yang, and C. Wang. Mobile data gathering with load balanced clustering and dual data uploading in wireless sensor networks. *IEEE Transactions on Mobile Computing*, 14(4):770–785, 2014.