An Efficient Approach for Storage Balancing in Wireless Sensor Networks

https://doi.org/10.3991/ijoe.v13i09.7090

Asmaa Ez-Zaidi(✉), Said Rakrak
Cadi Ayyad University, Marrakesh,Morocco
asmaa.ez.zaidi@gmail.com

Abstract—the use of mobile sinks in data collection has received much attention in recent years. In fact, mobility was introduced to solve problems that occur in data gathering with static sink such as hotspots, quick energy depletion of sensor nodes and so on. Using a mobile sink provides an effective mechanism to improve reliability, security as well as connectivity within the network. Nevertheless, the sink's mobility poses new challenges, especially when the sink follows an unpredictable movement while gathering data. In this case, the network will experience huge latency and suffer from significant packet loss particularly when sensor nodes do not have enough memory storage to buffer collected data between two consecutive visits of the mobile sink. In this paper we propose a new approach in which sensor nodes cooperate to manage the storage and prevent packet drops. When a node's memory is almost full, it offloads its data to its neighbor nodes in function of their free spaces. In case there are no suitable neighbor nodes with sufficient storage space, the sink is urgently notified about the overloaded region that needs to be rapidly dumped. Simulation results reveal that our proposed approach balances the storage load and decreases drastically packet loss.

Keywords— wireless sensor network; packet loss; distributed data storage; data gathering; data repartition; buffer overflow

1 Introduction

In recent years, wireless sensor networks (WSNs) have been the subject of intense research especially with the proliferation in Micro-Electro-Mechanical Systems (MEMS) technology. These networks are composed of a large number of sensor nodes that work together to monitor physical phenomena such as humidity, pressure, heat, light and so on. WSNs are used in several fields such as intelligent transportation systems [1], structural health monitoring [2], industrial applications [3], smart homes [4, 5] forest monitoring [6], localization [7] and many more interesting applications.

In typical wireless sensor networks, data is disseminated according to a multi-hop fashion towards a static sink. However, nodes located in the sink's vicinity drain their energy earlier because they relay more traffic than the other nodes leading to the so-

called hotspot problem [8]. Sink mobility [9] [10] was introduced in several works as a potential solution to alleviate the aforementioned issue and to balance the energy in the network. With sink mobility, data is collected via one-hop communication and the responsibility of delivering data to the sink is shared among sensor nodes. The motion of the sink can be random, controlled or predictable [11].

Using mobile sinks in data collection brings many advantages such as high reliability, an increase in network lifetime; more security and less transmission errors. However, it poses significant challenges, most notably data loss due to storage constraints of sensor nodes.

In fact, loss of data is not bearable in many sensitive applications such as civil structure health monitoring or target tracking [12] because it affects the quality and reliability of collected information. Reducing data loss caused by the lack of sufficient storage in data collection is a big challenge. Several strategies, as depicted in the related work section have been proposed by the research community to alleviate this issue.

Sensor nodes are well known by their inability to store large amount of data because of their limited storage. Thus, once their buffers fill up, they start dropping packets. This situation occurs when the mobile sink visits some parts of the network more frequently than others to collect data. Furthermore, the sink and sensor nodes have very short time to communicate when they are within radio range of each other, which means that only a portion of resources will be gathered. As a consequence, nodes have to wait till the sink returns back to complete the rest of data. At this stage, nodes may experience memory shortage as well as data loss particularly when the sink makes long journeys before returning back.

We try through this paper to address the problem of storage and prevent data loss. Sensor nodes collaborate with each other by splitting data proportionally. Nodes having large space memory, buffer data of nodes whose storage are running out. This process makes the network balanced and more uniform. The main objective of cooperative storage is to manage the available storage in the network to ensure the constancy of data collection for the longest possible without any disruption or data drop. Moreover, when a region reaches its limit, which means there is no possibility for data sharing because all the neighbor nodes are full, the mobile sink is notified through an emergency message. The latter moves towards the burdened area and start offloading data before node's buffer overflows.

The rest of the paper is organized as follows. Section 2 summarizes some of related work with regard to storage management techniques in wireless sensor networks, section 3 describes our approach; section 4 presents our simulation results. The last section concludes the paper.

2 Related work

Lot of research has been done on data storage management in wireless sensor networks. Some works have used data replication that consists in copying data at other

nodes to address the problem of node failures while others have used data distribution to balance the network storage and decrease the occurrence of packet drops.

Tseng et al. [13] proposed DSM a distributed storage management strategy that aims to reduce data drop due to node's limited memory space and to avoid the loss of high priority data in an isolated network.

Authors in [14] proposed an approach to solve issues related to buffer overflow as well as packets beyond deadlines. They adopted a hybrid data gathering algorithm that mixes locally passive and globally proactive uploading. Upon moving, the sink stops at each Rendezvous point (the virtual root of K-hop spanning tree) to collect data. When a node's memory is about to overflow and could not find available sensor nodes in the tree (parent node, child node, neighbor node in the same sub-tree) or in a neighbor tree to share memory resources with; it proactively uploads packets towards the sink after calculating its current location based on a synchronized clock and a location-time function to avoid data loss.

In order to increase the resilience and overcome the problem of node failures, authors proposed in [15] a distributed data replication mechanism that aims at distributing N replicas of each data generated by a node across the wireless sensor network. To do so, each sensor node designates from its neighbors a donor node that has large memory availability to store a replica of data. Once the donor node receives the copy of data, its stores it in its local memory and chooses among its neighbors the next donor node, this process is repeated until the number of replicas is reached. A sensor node drops the acquired data when its local memory is full and could not find a suitable donor.

In [16] authors proposed a Qos data collection approach in which the tour of the mobile sink is planned in such a way that data is collected from sensor nodes before their storage space becomes full.

Authors in [17] proposed dissemination techniques to store data items in communicating materials. In localized dissemination, data is stored in a limited region of the material. Packets are forwarded from one node to another, each node that receives the packet stores the information and decrement the hop counter, and so on until reaching zero. In non-localized dissemination, data is replicated in a uniform manner which facilitates its retrieval from any piece of the material even after shape transformation.

A cooperative caching scheme named ZCS for wireless sensor networks was introduced in [18] by Narottam Chand. In order to find the node who has stored the queried data, ZCS employs a discovery algorithm. One-hop neighbors of a given sensor node share cashed data from one another to form a cooperative cache system zone. For each data request, the sensor node first checks its local cash, if data item is valid the query is served, otherwise the node checks if the data is cashed within its home zone. In case of a zone cash miss, data is searched in nodes belonging to other zones. Finally, if the data is not found in remote zones, data is retrieved from the source node.

Lin et al. [19] addressed both the energy consumption and buffer overflow problems in data collection. In their approach sensor nodes are clustered by constructing a dominating set, which are then assigned as rendezvous points. They also presented an

allotment mechanism in which sensor nodes having a high data sampling rates buffer their data in the memory of sensor nodes with a lower sampling rate in order to mitigate the buffer overflow issue.

Authors in [20] discussed the k-storage node problem which consists in selecting the adequate k-nodes as storage nodes in the network. To do so, they proposed three schemes which are data storage scheme based on random strategy, data storage scheme based on greedy strategy and data storage scheme based on evolutionary algorithm strategy.

Authors in [21] proposed Proflex a protocol that exploits powerful sensor nodes to perform distributed data storage in heterogeneous wireless sensor networks. This protocol benefits also from the higher communication range of these nodes and uses the long range links to enhance data replication and distribution.

Our concern in this paper is to distribute data load among nodes in such a way that overloaded nodes send their extra data to underloaded ones in order to balance the network storage. Through this work, we will answer some challenging questions such as:

- How to reduce packet drops and delay its occurrence in the network.
- How to distribute data load among sensor nodes to balance the storage in the network.

3 Our proposed solution

3.1 System model

We consider a wireless sensor network (as shown in figure 1) that consists of a set of static sensor nodes deployed randomly within an area, and a mobile sink with unlimited resources that moves at a fixed speed around the field and gathers data on the fly.

Sensor nodes are aware of their positions and have the same communication range as well as storage space. They monitor periodically the region of interest, generate packets and buffer them locally while awaiting the arrival of the sink that moves in a random manner to collect data. The latter, changes randomly its movement direction and angle while crossing the network.

In our approach we used sink mobility for its ability to share data load among all sensor nodes within the network and to ensure a high reliability in the data collection process. While moving within the field, the mobile sink periodically broadcasts beacon messages to inform sensor nodes about its presence. Nodes having received the beacons upload their buffered data to the sink via one hop communication.

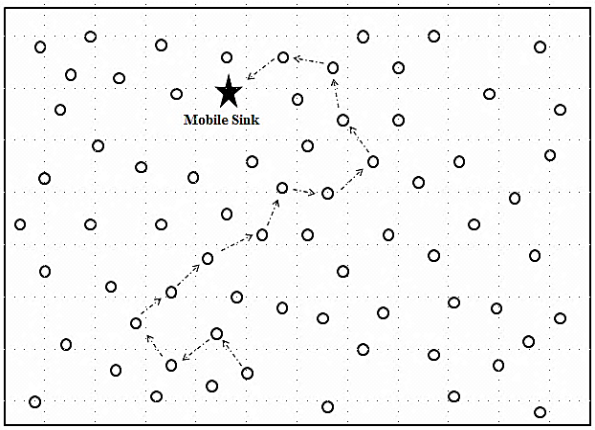


Fig. 1. Our network model.

3.2 Data partitioning

Sensor nodes buffer their measurements locally and wait the arrival of the mobile sink to collect it. But when the sink's trip is too long and crosses the network randomly (which means that some areas will be visited more than others) with a very low speed, these nodes may experience buffer overflow rapidly since they have very limited storage capacity and cannot hold data for a long time. Consequently, sensor nodes start dropping packets. Under this situation, the quality as well as reliability of collected data is degraded.

In order to mitigate this issue sensor nodes collaborate to store data in a distributed manner, highly loaded nodes shift their data towards underloaded ones. Doing so, the load will be distributed across the network and each sensor node will be able to store data for a long time, thing that will delay the occurrence of data loss.

Figure 2 depicts the scenario of data sharing among nodes. For example, the amount of free space in node 7 has dropped below a predetermined threshold. At that point, it must share its data before it becomes totally full and start dropping packets. To prevent this situation, it delegates storage to its neighbor nodes. To do so, node 7 asks its one hop neighbors for their free storage.

Upon receiving the query, the neighboring nodes calculate their available storage according to Equation 1 and respond the node 7 (the overloaded node).

Available storage = [Total_Capacity_Storage * (1-Threhsold)] - Occupied_Storage

Equation 1

- *Threshold* is a user-adjustable parameter that can change according to the network's environment. This threshold indicates a storage margin (set in our case to 10%) that a node should not declare in order to prevent it from becoming full when it receives the data from other full nodes.
- *Occupied_Storage*: Amount of data currently stored in the sensor node.

Finally, node 7 calculates the sum of the free storage space of all its neighbor nodes and compares it with the quantity of data it buffers locally. It then decides whether it will distribute all its data, or only a small portion of it.

Case 1: If the total available storage space of neighbor nodes (Ts) is less than data stored in node 7, the latter offloads only the two-thirds of Ts. Without this mechanism, after data sharing neighbor nodes might fill up their buffers, and thus they will send back their data to node 7.

Case 2: If the total available space of neighbor nodes is greater than data buffered in node 7, the latter offloads all its data to its neighbors in a proportional manner according to their free storage (Algorithm 1). Doing so, data will be fairly distributed among sensor nodes i.e. node1, node2, node3, node4 and node6.

Node 5 was excluded because it has reached an alarming threshold and thus it will not be able to accept new data, otherwise it will undergo a storage overflow. In this case, node5 will be forced to seek for potential neighbor nodes that can cache its data.

```
Set SumFreeStorage = Σ AvailableNeibhors(i).FreeStorage ;
If ( Node(i).UsedStorage < SumFreeStorage )
        Set DataToSend = Node(i). UsedStorage;
Else
        Set DataToSend = 2/3 * Node(i). UsedStorage;

Set scale = DataToSend / SumFreeStorage;

For (i = 0; i < availableNeighbors.size; i++) {
  Set currentNeighbor = availableNeighbors[i];
 currentNeighbor.scaledFreeStorage = scale * currentNeighbor.FreeStorage;

}
```

Algorithm 1. Proportional data distribution

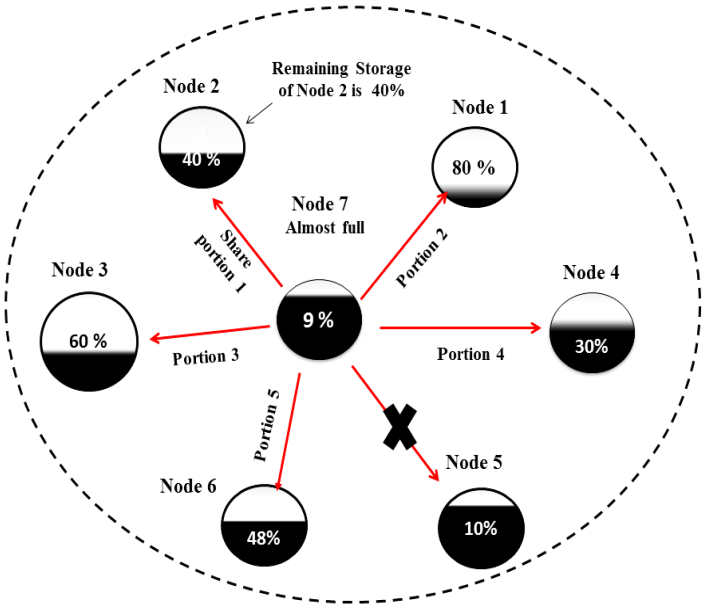


Fig. 2. Data repartition scheme

3.3 Sink notification

While moving, the sink broadcasts periodically beacon messages to advertise its presence. These beacons include the ID of the sink, its position and the current timestamp. Sensor nodes having received the beacon, update their table by setting to true a flag IN_VICINITY that indicates that are in direct contact with the mobile sink and can forward their buffered data if any.

When a node's remaining storage space is exhausted, and could not find suitable neighbor nodes with sufficient memory to share data with. It notifies the mobile sink through an urgent message (REQUEST_FULL_REGION) about the overloaded region that has to be rapidly dumped. The structure of the REQUEST_FULL_REGION message contains the coordinates of the full region, the ID of the source node that emitted the request and a TTL flag.

In order to locate the current position of the sink within the network, the full node sends a REQUEST_FULL_REGION to its one-hop neighbors (TTL = 1) to ask them if the sink is in their vicinity (figure.3a). Then it starts a timer waiting for neighbors' replies. If the timer expires, and the full node did not receive any reply, it will consider that none of its one-hop neighbors has seen the mobile sink (Algorithm 2a).

```
Set  TTL = 0 ;
Set  Receive_Reply = false ;
While ( ! Receive_Reply) {
        TTL++;
        Broadcast_FullRegion_Request  (TTL) ;
        Launch_Timer (TTL * Tdata_Forward);
}
```

Algorithm 2a. Localize the sink's current position (Executed by the overloaded node)

Receive_Reply is set to true once the overloaded node (node 7) receives a reply from nodes having the sink nearby.

Tdata_Forward : is the time needed by a node to forward a packet to another node.

```
When Node(j) receives a REQUEST_FULL_REGION message {
      If (Node(j). IN_VICINITY == true) {

// inform the overloaded node that the request has been delivered to the sink
(Set Receive_Reply = true).
        Node(j).SendReplyToSourceNode();
        Node(j).ForwardRequestToSink();    // inform the sink about the full
region.
   }
      else {
              TTL = ExtractTTLFromMessage ();
              TTL-- ;
             If (TTL > 0)
              Node(j). Broadcast_FullRegion_Request  (TTL) ; // Forward
REQUEST_FULL_REGION message to the nodes in the next hop.
              else
              DropPacket();
      }
}
```

Algorithm 2b Localize the sink's current position (Executed by neighboring nodes)

The full node will then increase its search area by sending a REQUEST_FULL_REGION to its two-hop neighbors, by incrementing the TTL value by 1, to seek for the sink (figure 3b). This process is repeated until finding the exact location of the mobile sink.

Sensor nodes forward the urgent message to the sink when they are within range of each other i.e. IN_VICINITY flag = true (figure 3c). In the same time, they inform the full node that the sink has successfully received the request message (Algorithm 2b).

Upon receiving REQUEST_FULL_REGION, the mobile sink moves immediately towards the congested region to start data offloading. It stops for an appropriate interval in function of the amount of data to offload from that region. Once the offload is finished, the sink consults its table to see which region will visit next.

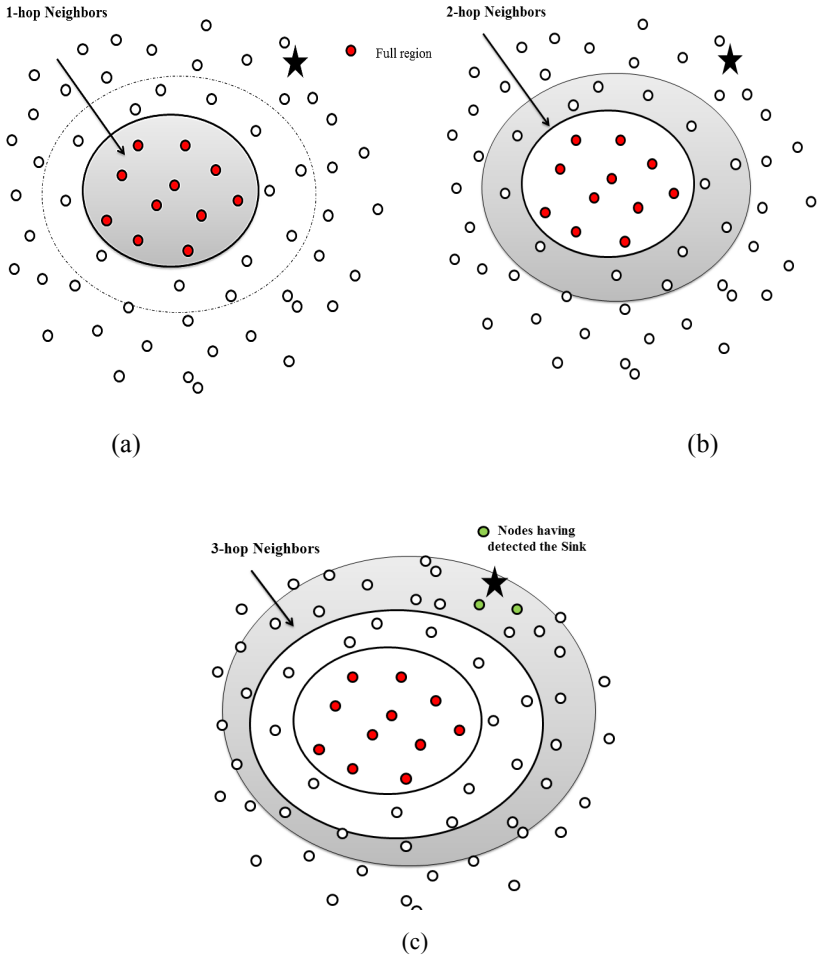
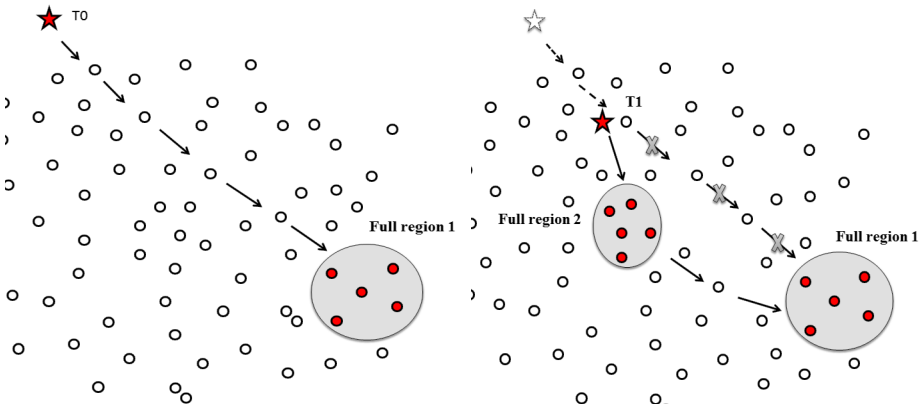


Fig. 3. Sink discovery in full region scenario.

There are some cases in which the mobile sink receives multiple requests from other full areas while moving to offload a predefined one (figure 4a). In this case, the mobile sink checks whether the position of the other regions are in its direction. If it is the case, it will first serve these regions (figure 4b) and then visits the current one. Otherwise, the mobile sink will add the requests to its queue table to visit them later.



(a) The sink moves toward full region 1 to offload data - **(b)** The sink receives another request from region 2.

Fig. 4.

4 Experimental evaluation

In this section, we evaluate the performances of our proposed approach via NS-2.35 simulator. We have considered a network area of 250 m * 250 m in which it is deployed 50 sensor nodes. All nodes have the same communication range R= 50m and the same storage capacity. We consider also, a sink moving at a constant speed that crosses the network in a random manner to collect data from sensor nodes.

Table 1.

Simulation parameters	Values
Number of mobile sink	1
Number of nodes	50
Simulation area (m2)	250 * 250
Simulation time (scd)	4000
Speed of the sink (m/s)	4
Communication range (m)	50
Sensing rates (data unit /scd)	64, 92,128,160,192
Mac Protocol	802.15.4
Channel	Wireless
Propagation	Two-Ray-Ground
Packet size (byte)	32

We evaluate our approach according to different parameters which are data loss percentage, the time when the first node fills up its local memory and the total full nodes during the whole simulation. The simulation has been run several times (50 times). For each simulation scenario we took the average of runs as final results.

4.1 The time of the first full node appearance

In this section, we computed the time of the first full node appearance in the network under different rates: 64, 92, 128, 160 and 192 data unit/second. We have compared our approach that uses the concept of data sharing among nodes and sink notification with a classical approach in which the sink simply crosses the network randomly and collects data. We aim through this simulation to show the strength of our scheme.

From Fig. 5 and without using our approach, we notice that data loss appears earlier (exactly in 829 seconds) for the rate 64 data units/sec. For rate 192, sensor nodes run out of storage space rapidly (240 seconds) and start dropping packets. Conversely, we can see that our strategy has notably delayed the occurrence of data loss in the network from 829 seconds to 1148 seconds for the rate 64. This goes back to the fact that the loaded nodes share their resources with their underloaded one-hop neighbors, which creates a certain balance in the network. The sink's notification to gather data has also played a key role in extending the time of data loss appearance.

4.2 Packet loss

Packet loss: Represents the quantity of dropped packets in the network. As shown in figure 6, although the rate is high (case 192 data unit / second), dropped packets in our approach is low (19 %) compared with the classical approach (63%). This is goes back to the fact that when sensor nodes reach a predetermined threshold, they delegate storage to their one-hop neighbor nodes that have enough available space. Moreover, when the sink receives a notification request from a region that runs out of storage space, it immediately moves towards it to gather data before any data loss occurs. Conversely in the classical approach, when sensor nodes face storage space shortage, they can no longer collect or buffer data locally thus they start dropping packets.

4.3 Number of full nodes

Packet loss and number of full nodes are connected with each other. As the number of full nodes increases in the network, the number of dropped packets increases too. According to the graph below, our approach outperforms the classical one; we can see that the number of nodes that have run out of storage space is very low. It varies between 2 and 8, whereas in the classical approach the number of full nodes varies between 11 and 20 (Fig. 7).

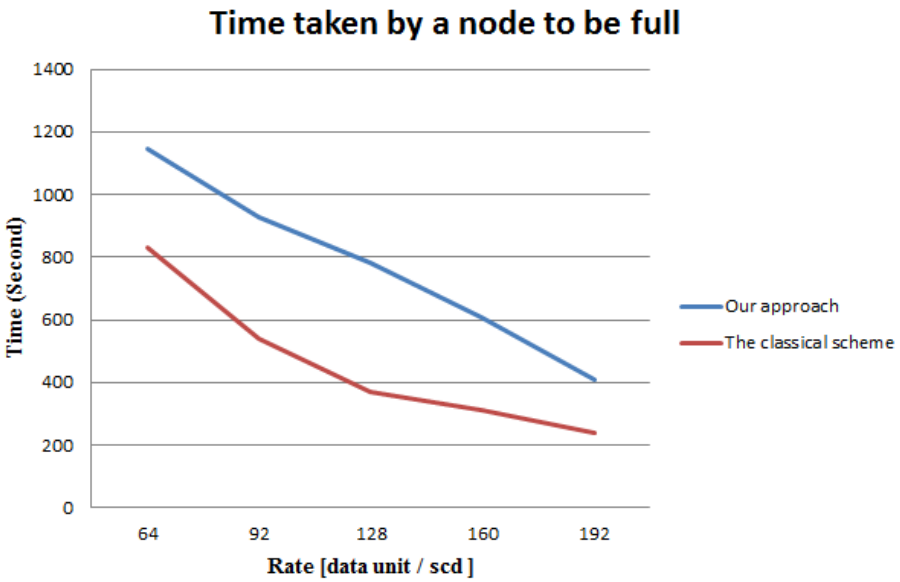


Fig. 5. Time taken by a node to full

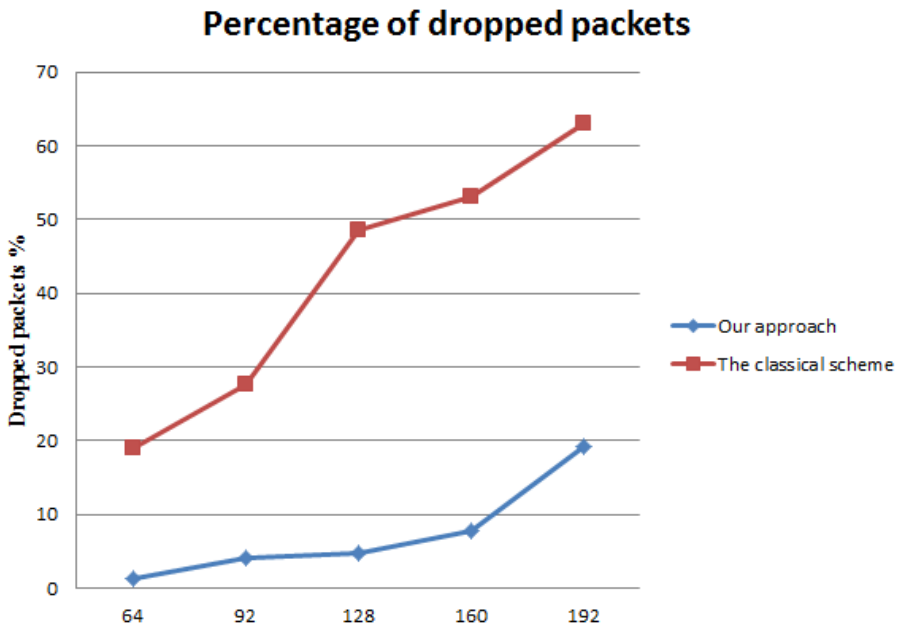


Fig. 6. Percentage of dropped packets (%)

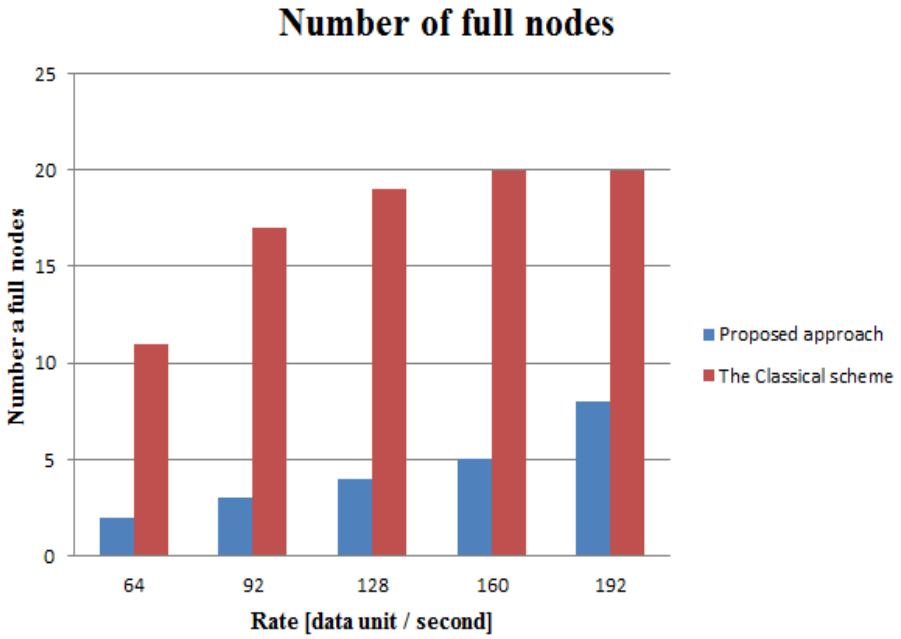


Fig. 7. Number of full nodes at the end of simulation

5 Conclusion

In this paper, we have presented a new approach that mitigates the issue of data loss in data collection and maximizes nodes' storage capacity utilization. The cooperative effort of sensor nodes helps in delaying the occurrence of data loss caused by the limited memory of nodes. The large storage reserves of some nodes in the network have been exploited by those that are running out from storage which maximize node's storage utilization.

Furthermore, sending alerts to the sink in order to dump data from loaded regions will no doubt decrease packet drops.

Simulation results reveal that our approach reduces drastically packet drops caused by quick buffer overflows. Thanks to the proportional data distribution among sensor nodes and sink notification strategies, the sensor network becomes more balanced and uniform. As a future work we will address the issue of energy consumption in the network, sensor nodes use lot of energy in data transmission which impacts their battery lifetime.

6 References

[1] Anisi, M. H., & Abdullah, A. H. (2016). Efficient data reporting in intelligent transportation systems. Networks and Spatial Economics, 16(2), 623-642. https://doi.org/10.1007/s11067-015-9291-9

[2] Noel, A., Abdaoui, A., Badawy, A., Elfouly, T., Ahmed, M., & Shehata, M. (2017). Structural Health Monitoring using Wireless Sensor Networks: A Comprehensive Survey. IEEE Communications Surveys & Tutorials. https://doi.org/10.1109/COMST.2017.2691551

[3] Ovsthus, K., & Kristensen, L. M. (2014). An industrial perspective on wireless sensor networks—a survey of requirements, protocols, and challenges. IEEE Communications Surveys & Tutorials, 16(3), 1391-1412. https://doi.org/10.1109/SURV.2014.012114.00058

[4] Ransing, R. S., & Rajput, M. (2015, January). Smart home for elderly care, based on wireless sensor network. In Nascent Technologies in the Engineering Field (ICNTE), 2015 International Conference on (pp. 1-5). IEEE.

[5] Ghayvat, H., Mukhopadhyay, S., Gui, X., & Suryadevara, N. (2015). WSN-and IOT-based smart homes and their extension to smart buildings. Sensors, 15(5), 10350-10379. https://doi.org/10.3390/s150510350

[6] Molina-Pico, A., Cuesta-Frau, D., Araujo, A., Alejandre, J., & Rozas, A. (2016). Forest Monitoring and Wildland Early Fire Detection by a Hierarchical Wireless Sensor Network. Journal of Sensors, 2016. http://dx.doi.org/10.1155/2016/8325845.

[7] Qiu, Z., Wu, L., & Zhang, P. (2017). An Efficient Localization Method for Mobile Nodes in Wireless Sensor Networks. International Journal of Online Engineering, 13(3). https://doi.org/10.3991/ijoe.v13i03.6868

[8] Jannu, S., & Jana, P. K. (2016). Energy Efficient Algorithms for Hot Spot Problem in Wireless Sensor Networks. In Proceedings of the Second International Conference on Computer and Communication Technologies (pp. 509-517). Springer India. https://doi.org/10.1007/978-81-322-2517-1_49

[9] Liao, W. H., & Kuai, S. C. (2012, September). An Energy-Efficient Data Collection Mechanism with a Mobile Sink for Wireless Sensor Networks. In Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing (UIC/ATC), 2012 9th International Conference on (pp. 210-216). IEEE. https://doi.org/10.1109/UIC-ATC.2012.130

[10] Di Francesco, M., Das, S. K., & Anastasi, G. (2011). Data collection in wireless sensor networks with mobile elements: A survey. ACM Transactions on Sensor Networks (TOSN), 8(1), 7. https://doi.org/10.1145/1993042.1993049

[11] Khan, M. I., Gansterer, W. N., & Haring, G. (2013). Static vs. mobile sink: The influence of basic parameters on energy efficiency in wireless sensor networks. Computer communications, 36(9), 965-978. https://doi.org/10.1016/j.comcom.2012.10.010

[12] Ez-Zaidi, A., & Rakrak, S. (2015). A comparative study of target tracking approaches in wireless sensor networks. Journal of Sensors, 2016.

[13] Tseng, Y. C., Wu, F. J., & Lai, W. T. (2013). Opportunistic data collection for disconnected wireless sensor networks by mobile mules. Ad Hoc Networks, 11(3), 1150-1164. https://doi.org/10.1016/j.adhoc.2013.01.001

[14] Zhu, C., Han, G. J., Wang, Y., Bao, P., & Shen, J. (2016). A Memory Sharing based Low Loss Rate Data Gathering Algorithm with a Mobile Sink for Wireless Sensor and Actuator Networks. Asian Journal of Control.

[15] Gonizzi, P., Ferrari, G., Gay, V., & Leguay, J. (2015). Data dissemination scheme for distributed storage for IoT observation systems at large scale. Information Fusion, 22, 16-25. https://doi.org/10.1016/j.inffus.2013.04.003

[16] Liao, W. H., & Kuai, S. C. (2015, July). A QoS data collection scheme for wireless sensor networks. In Ubiquitous and Future Networks (ICUFN), 2015 Seventh International Conference on (pp. 612-617). IEEE. https://doi.org/10.1109/ICUFN.2015.7182617

[17] Mekki, K., Derigent, W., Rondeau, E., Zouinkhi, A., & Abdelkrim, M. N. (2013, October). Multi-hop counter based data dissemination protocol for communicating materials. In

Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on (pp. 45-52). IEEE. https://doi.org/10.1109/WiMOB.2013.6673339

[18] Chand, N. (2012). Cooperative Data Caching in WSN. World Academy of Science, Engineering and Technology, 63, 90-94.

[19] Lin, P. L., & Ko, R. S. (2011). An efficient data-gathering scheme for heterogeneous sensor networks via mobile sinks. International Journal of Distributed Sensor Networks.

[20] Guo, J., Sun, L., Han, C., & Liu, L. (2017). K-Storage-Node Problem of Distributed Data Storage for Internet of Things. Intelligent Automation & Soft Computing, 1-8. https://doi.org/10.1080/10798587.2017.1316068

[21] Maia, G., Guidoni, D. L., Viana, A. C., Aquino, A. L., Mini, R. A., & Loureiro, A. A. (2013). A distributed data storage protocol for heterogeneous wireless sensor networks with mobile sinks. Ad Hoc Networks, 11(5), 1588-1602. https://doi.org/10.1016/j.adhoc.2013.01.004

7 Authors

Asmaa Ez-zaidi (corresponding author) received her Diploma in Computer Engineering from Cadi Ayyad University. She is currently a PhD student at the faculty of science and technology, cadi Ayyad University. She is also a member at the Laboratory of Applied Mathematics and Computer Science (LAMAI) Marrakech, Morocco. Her research interests are in the areas of wireless sensor networks. (asmaa.ez.zaidi@gmail.com).

Said Rakrak is a professor in the department of computer science at the faculty of sciences and technology, Cadi Ayyad University. He is also a member at the Laboratory of Applied Mathematics and Computer Science (LAMAI) Marrakech, Morocco. His research interests are wireless sensor networks, RFID, cloud computing and Internet of things (IOT).

Article submitted 01 May 2017. Published as resubmitted by the authors 13 June 2017.