CORE: Coordinated Relocation of Sink Nodes in Wireless Sensor Networks

Jesse English, Michael Wiacek and Mohamed Younis

Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County
(jesse.english, mw, younis@umbc.edu)

Abstract – ***In wireless Sensor Networks (WSNs), collected data are routed to multiple gateway nodes for processing. Based on the gathered reports, the gateways may also work collaboratively on a set of actions to serve application-level requirements or to better manage the resource-constrained WSN. Some of these actions involve the relocation of some gateways. In this paper, we argue that changing the position of a gateway cannot be pursued without the consideration of the impact on inter-gateway connectivity. We present an efficient algorithm for Coordinated Relocation of gateways (CORE). CORE strives to maintain communication paths among gateways while repositioning individual gateways to better manage the sensors in their vicinity. Simulation results have demonstrated the effectiveness of CORE and its positive effect on both network longevity and node coverage.***

1. Introduction

Wireless sensor networks (WSNs) have numerous applications in a variety of disciplines, both military and civilian [1][2]. The ability to remotely measure ambient conditions and track dynamic targets/events can be invaluable; especially in harsh environments where human intervention is risky or infeasible. Sensors are usually battery-operated and have a limited transmission range and onboard processing capacity. Such constraints have motivated lots of research on effective management strategies of WSNs that trade off resources, data fidelity, latency, and coverage so that the network can stay functional for the longest duration.

A typical WSN architecture involves large numbers of sensor nodes that report their measurements to locally deployed data collection centers; often referred to as sink or gateway nodes. Gateways are usually more capable in terms of their energy supply, radio range and computational resources. In many applications, gateways coordinate among themselves and even move around to serve the application. For example, in a disaster management setup multiple robots may be deployed to collect in-situ sensor data, analyze such data and collaboratively perform a rescue mission. Another example is the use of rovers on planetary and lunar surfaces.

By their very nature, WSNs should be both ad-hoc and scalable. Physical deployment of a WSN should not have to be planned; instead nodes should be capable of being dropped in an area, and self-organizing. To support scalability, sensors are often grouped into clusters; each is led by a gateway node. Within a cluster, sensors reports are forwarded over multiple hops to the respective gateway node, which processes the data and then deliberates with the other gateways on the right course of actions. Therefore, a gateway node should not only reach the sensors in its cluster but also maintain communication channels to other gateways.

The location of the gateway can be very influential to the network performance. For example, routing data to a gateway that is distant from the source sensors usually involves numerous relaying nodes and thus increases the aggregate delay and energy consumption and risks a packet loss due to link errors. In addition, ill-placed gateway nodes may cause some of them to be isolated from the rest and just prevent them from conducting the level of coordination required by the application. Gateway relocation has been shown to be an effective optimization strategy that boosts network longevity, timeliness and reliability [3][4]. However, most of the published schemes consider only a single gateway scenario. We argue that limiting the scope of the analysis to the intra-cluster network can cause network partitioning and diminish the potential of inter-gateway coordination.

In this paper we present CORE; an algorithm for Coordinated Relocation of gateway nodes. CORE considers the inter-cluster implications of a gateway repositioning and derives the necessary conditions for approving a gateway move. Even if the motive for a gateway's relocation may be local to its cluster, the global effect of such a change in position is categorized and the appropriate actions are performed. In case a strongly connected inter-gateway topology can be achieved, CORE allows the repositioning. To handle the possibility of creating a partitioned inter-gateway topology, CORE pursues novel heuristics that trigger the relocation of multiple gateways in order to improve the operation within individual clusters. To the best of our knowledge, the problem of coordinated repositioning of multiple gateway nodes in WSNs has not been addressed in the literature.

This paper is organized as follows. Section 2 reviews related work in the literature. We describe the system model and state our assumptions. In Section 3, we describe the problem and enumerate a variety of scenarios that can exist when moving sink nodes in a multi-cluster environment. In Section 4, we discuss the CORE algorithm in details. Validation results are presented in Section 5. Finally, Section 6 concludes the paper.

2. Related Work

Base-station (gateway) repositioning has been investigated in the context of wireless local area networks and cellular infrastructure [5]. The base stations, in these systems are stationary in nature and are placed in order to achieve coverage of an area or a building using a minimal number of base stations. Another related work is reported in [6]. The considered model is to use a gateway node as a direct router for a group of mobile nodes that would be otherwise

unreachable due to topological reasons such as blockages. The problem addressed is to find the optimal place for the gateway node to best serve the group in terms of latency and throughput. The pursed approach is to move the gateway to the weighted geographic centroid of the group by considering the location and traffic generated by nodes regardless the established routes. The approach of [3] follows similar methodology but it is path-based. They argue that it is not possible to optimally place the gateway without considering the network topology and inter-node links. None of these efforts addressed the collaborative relocation of multiple inter-connected gateways.

Optimal placement of gateway nodes has been studied in [7][8][9][10]. The scope of [7][8] is limited to a single gateway node. On the other hand, the approach of [9] strives to maximize the data flow in the network by carefully positioning multiple gateway nodes. To counter the NP-complete nature of the placement problem [8], the authors proposed two different approximation algorithms. The aim of [10] is to extend the network lifetime by distributing the load among the deployed sensors. An integer linear programming formulation was proposed to promote fairness among sensor nodes in terms of data flow. However, this type of research considers only static metrics and is applied at the time of network setup. We are not aware of any work that addresses inter-gateway networking issues while repositioning the individual gateways. Currently we are extending CORE to handle inter-cluster communication via relay sensors.

3. System Model

We assume that during the network bootstrapping phase the gateways will perform node discovery to establish contact with the largest number of deployed sensors [11]. The gateways then divide the discovered sensor nodes among them, forming clusters. Our approach is independent from the clustering criteria which may be load balancing, communication range, etc. [12]. A gateway is responsible for managing the sensors in its clusters. A sensor node is capable of communicating with the gateway of its cluster either directly or over a multi-hop path. Nodes that are incapable of reaching any gateway are considered orphaned.

A gateway node is assumed to have motion capabilities, e.g a rover. We assume that gateways understand the layout of their own clusters in relative terms at least. We also assume that a gateway node has initial contact with all other gateways in the network. Our algorithm will maintain this contact, which will ensure that gateway nodes (through inter-gateway communication) will be able to build a global view of the entire network by sharing local views of clusters. Gateway nodes do not need to be in direct transmission range of all other gateways in the network, but a path must be available that links each gateway to another. Fig. 1 shows a sample valid network setup.

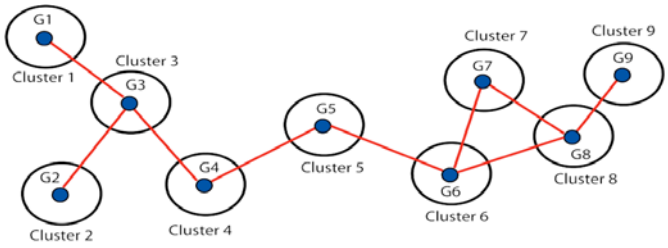

Fig. 1: Gateways have communication path to each other

4. Problem Description

As indicated earlier, gateway relocation is an effective scheme for optimizing the network operation within the individual cluster. The main issues are when it makes sense to move the gateway, where to place it and how to handle the packet traffic while the gateway is on the move [3]. However, the relocation problem is substantially compounded when a network consists of multiple clusters. A gateway node cannot choose to arbitrarily wander around its cluster to enhance the intra-cluster network operation (local gain) without considering the potential impact this could inflict on its ability to maintain communication with the gateway nodes of other clusters (global effect). In this context, we define local gain as improvement to the number of sensor nodes in the cluster (coverage), the average hops required to communicate with the sink node from a sensor node (cost of communication), and throughput of data to the sink node – all other relevant data can be determined from these metrics.

In terms of gateway-to-gateway communications, at the coarsest grained level, when a gateway node moves one of three things can happen. Links between gateway nodes can be broken, links between gateway nodes can be formed, or no change to the network occurs. When a link is broken between two gateway nodes, e.g. G_5 and G_6 in Fig. 1, the resulting severed communication could affect the remaining gateways. In the configuration of Fig 1, breaking the link G_5G_6 causes a partitioning in the inter-gateway network (preventing G_1, G_2, G_3, G_4 and G_5 from reaching G_6, G_7, G_8 and G_9)

In addition, uncoordinated relocation of individual gateways can raise race conditions. Assume that a gateway, say G_4, is careful when deciding on its new location such that it stays reachable to all gateways to which it has communication links (G_3 and G_5). Even with the precautions that G_4 takes, a network partitioning can still occur when G_3 and/or G_5 simultaneously relocate. The reason is that G_4 will base its relocation process on positions of G_3 and G_5 which becomes invalid when they decide to move. Therefore, a synchronization protocol needs to be applied in order to ensure that a gateway bases its relocation decision on a consistent global state; particularly the location of all gateways.

5. The CORE Algorithm

CORE strives to counter the multi-gateway relocation issues discussed in the previous section. The goal is to enable optimization of the intra-cluster operation while ensuring the inter-gateway connectivity. Thus, the first phase of CORE involves locating a suitable place for a gateway node G to

relocate to in order to better serve its cluster. A number of schemes are proposed in the literature for finding such new location [3][4]. A scenario is depicted in Fig. 2. Upon identifying the new location, CORE requires *G* to base its analysis on a consistent network state. That is being implementing by using a *mutex*. Basically, a global variable is defined and shared among the gateways (we are in the process of removing the need for this shared data resource by using distributed synchronization techniques). Access to this *mutex* is exclusive and granted to only one gateway at a time. Such exclusive access to this *mutex* will allow only one gateway to restructure the inter-gateway topology and prevent the potential of race conditions discussed earlier.

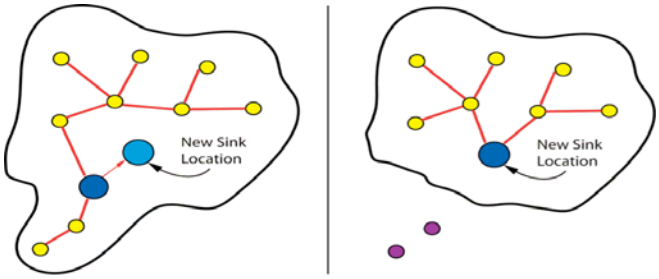

Fig. 2: The movement of the gateway node restructures the intra-cluster topology, leaving a few source nodes orphaned.

While having the exclusive access to the *mutex*, *G* is sure that there is no change in the network state and can thus go forward to evaluate the inter-cluster ramifications of its move. If the move is deemed appropriate, i.e. it does not cause partitioning of the inter-gateway network, *G* can go ahead and relocate. *G* then releases the *mutex* and informs the other gateways about its new location. If, however, *G* finds that its move will sever the network, it will need to initiate a recursive process of requests for movement to attempt to keep the inter-gateway connectivity. To do this, *G* will have to examine exactly which communication links it would break by moving. For each of these links, *G* will need to make a request to the gateway node *G'* on the other end of the link. *G'* must then find a location to move to, in order to reestablish the link while maintaining good performance within its cluster. *G'* will strives to minimize the distance it travels in order to maintain other links it has. In case, *G'* loses connection with some of its neighbor gateways, it asks them to adjust their location and so on. To terminate the procedure, a gateway will be allowed to adjust its location only once.

At the functional level, the CORE algorithm consists of five parts. The outer CORE function, *EvaluateAndMove()* is responsible for initializing the recursion, maintaining the critical section, i.e. the *mutex*, and deciding to send the movement message. Inside *EvaluateAndMove()*, CORE needs access to the *FindNewLocation()* function. This function is responsible for identifying a desired location *inside* the gateway's cluster to move to. Following this, the *MoveToLocation()* recursive function is called. This function trickles down through the network of connected gateways,

```
EvaluateAndMove() {
   NewLocation = FindNewLocation()
   MUTEX:  BEGIN CRITICAL SECTION
   Initialize NetworkState
   NetworkState = MoveToLocation(NewLocation, me, NetworkState)
   IF (Links are broken) {
      FOR each GatewayNode in NetworkState {
         MoveToLocation (NewLocation GatewayNode. NetworkState)
   MUTEX: END CRITICAL SECTION

MoveToLocation(Location, SinkNode, NetworkState) {
   NetworkState.List.add(me)
   NetworkState.UpdateVirtualNetwork(me, Location)
   IF (Location breaks inter-cluster communication) {
      FOR each BrokenSink {
         IF (NetworkState.List does not contain BrokenSink) {
            HelpLocation = FindHelpLocation(me.Location, BrokenSink)
            NetworkState = MoveToLocation(HelpLocation, BrokenSink,
                                          NetworkState)
   RETURN NetworkState
```

Fig. 3: Outline of the CORE algorithm

recalling itself for each subsequent link break and updating CORE's view of the virtual network via the *UpdateVirtualNetwork()*. The recursion also needs one other function, *FindHelpLocation()*, which calculates the closest location for a gateway to move to establish communication with a given location subject to intra-cluster constraints. When the recursion returns, *G* will compare the view of the network with the view presented in the virtual network. Only if the virtual network performs better in simulation based on recent network activity, *G* will actually move.

6. Simulation Results

To simulate the effects of our approach on a network, we constructed a flexible simulator engine to handle a variety of test cases. The engine was designed to allow sensors and gateway nodes to be scattered over a field (up to 600). The gateway nodes would form clusters using communication cost metric [12], and then using Dijkstra's Shortest Path Algorithm, dispense a multi-hop communication graph through the cluster. Only one restriction was placed on the selection of locations for gateway nodes by requiring them to be within a maximum communication range of at least one other gateway node. This ensured that the network of gateways was not divided from the start.

After a network had been created, targets would be dropped in the area. Targets represent areas of interest, and could be anything from a moving vehicle, to a raging fire. In our simulations, targets used a variety of different movement algorithms: linear, random, and stationary. In addition, targets could also simulate growth, or increasing area of effect. This served to simulate an area of interest that spreads over time, such as a forest fire. A sensor for which a target is within its sensing radius sends a packet over multi-hop to the gateway. Each time a packet is transmitted the battery life of the sensor and receiver is adjusted. When a node reached zero battery life, it was considered dead, and the network was restructured around it.

Simulations were run on a large array of network parameters such as the initial battery life of a node, the cluster diameter in terms of number of hops from the gateway, target movement type, target placements, etc. Each experiment lasted for 5,000 cycles and was replicated 50 times over; each time a new random network using the specified parameters was generated. The simulation was run on that starting network configuration once with CORE and once without relocating the gateway at all. We applied the scheme of [3] to determine a new gateway location.

Due to space limitations, only a subset of the results from the experiments is shown below. Fig. 4 shows the average lifetime of sensor nodes. The figure confirms that in all cases the algorithm proposed extended lifetime of nodes on the average, regardless of the battery life of the nodes used. Obviously through the gateway relocation, CORE is making a gain in node lifetime in comparison to the baseline. It is also logical to look at the number of orphaned nodes at the end of each simulation. Recall that the two networks are exact duplicates, so any sensor node not originally discovered will never be used without CORE. Fig. 5 shows that up to 65% less sensor nodes are orphaned when using CORE.

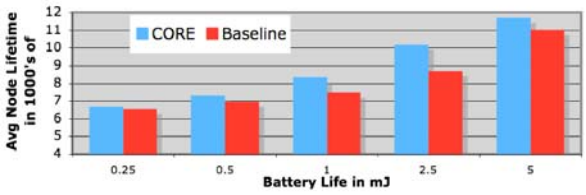

Fig. 4: The average lifetime of a sensor node is substantially increased no matter the battery life used

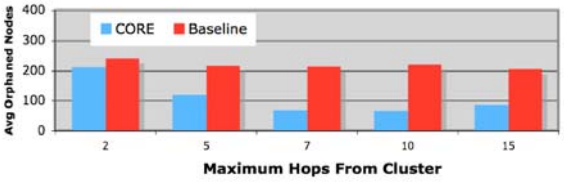

Fig. 5: The network coverage is dramatically boosted through the application of CORE (reduced number of orphaned nodes)

Fig. 6 shows the average number of packets received at each gateway through the course of the simulation. Coupling this with the data shown in Figures 4 and 5, it becomes apparent that CORE is not only saving energy, but also doing so while producing more data by boosting the network throughput.

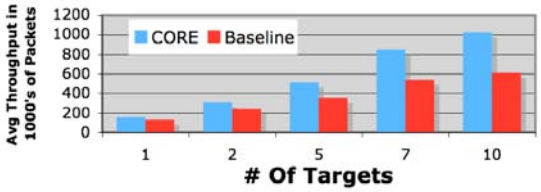

Fig. 6: The average number of packets received at a gateway node is substantially increased when CORE is applied.

7. Conclusion

In wireless sensor networks, data are collected at one or multiple gateway nodes for processing. In many application setups, gateways coordinate among themselves in order to efficiently and effectively handle the requirements of the application. In this paper we have shown that repositioning individual gateways can break inter-gateway communication links and thus risk the disruption of the network operation. We presented CORE a simple, yet effective, algorithm that tackles inter-gateway coordination issues. CORE checks the impact of relocating one gateway on the inter-gateway connectivity possibly triggering adjustments of the position of other gateways in order to maintain a strongly connected inter-gateway topology. CORE has been validated in a simulated environment of a target tracking application. Due to space constraints, the discussion on overhead analysis, and comparison to existing schemes has been omitted. The experimental results have demonstrated the effectiveness of CORE and its positive impact on contemporary metrics like network longevity and node coverage by allowing individual gateways more degree of freedom in optimizing their operation through relocation.

References

[1] I. F. Akyildiz, et al., "Wireless sensor networks: a survey", *Computer Networks*, Vol. 38, pp. 393-422, 2002.

[2] C-Y. Chong and S.P. Kumar, "Sensor networks: Evolution, opportunities, and challenges," *Proceedings of the IEEE*, 91(8), 2003.

[3] K. Akkaya, M. Younis and M. Bangad "Sink Repositioning for Enhanced Performance in Wireless Sensor Networks," *Computer Networks*, Vol. 49, pp. 512-434, 2005.

[4] Z. Maria Wang, S. Basagni, E. Melachrinoudis and C. Petrioli, "Exploiting Sink Mobility for Maximizing Sensor Networks Lifetime", in the *Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS'05)*, Big Island, Hawaii, January 2005.

[5] R. Rodrigues et al., "Optimal Base Station Placement and Fixed Channel Assignment Applied to Wireless Local Area Network Projects," *Proc. of IEEE International Conference on Networks*, Australia, 1999..

[6] M. Ahmed et al., "Positioning Range Extension Gateways in Mobile Ad Hoc Wireless Networks to Improve Connectivity and Throughput," *Proc. of IEEE Military Communications Conference (MILCOM 2001)*, Washington, D.C., October 2001..

[7] J. Pan, L. Cai, Y. T. Hou, Y. Shi, and S. X. Shen, "Optimal Base-Station Locations in Two-Tiered Wireless Sensor Networks" *IEEE Transactions on Mobile Computing*, Vol. 4, No. 5, 2005.

[8] A. Efrat, S. Har-Peled, J. S. B. Mitchell, "Approximation Algorithms for Two Optimal Location Problems in Sensor Networks" *Proc. of the 3rd International Conference on Broadband Communications, Networks and Systems (Broadnets 2005)*, Boston, Massachusetts , October 2005.

[9] A. Bogdanov E. Maneva, S. Riesenfeld, "Power-aware Base Station Positioning for Sensor Networks", in the Proceedings of the 23rd International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2004) Hong Kong, March 2004.

[10] H. Kim, Y. Seok, N. Choi, Y. Choi, and T. Kwon, "Optimal Multi-sink Positioning and Energy-efficient Routing in Wireless Sensor Networks" *Lecture Notes in Computer Science (LNCS)*, Vol.3391, pp.264-274, Springer-Verlag, January/February, 2005.

[11] R. Mathew, M. Younis and S. Elsharkawy "Energy-Efficient Bootstrapping Protocol for Wireless Sensor Network," *Innovations in Systems and Software Engineering*, 1(2), pp. 205 – 220, September 2005.

[12] G. Gupta, M. Younis, "Load-Balanced Clustering in Wireless Sensor Networks," in the *Proceedings of the IEEE International Conference on Communication (ICC 2003)*, Anchorage, Alaska, May 2003.