Algorithmic Construction of Optimal and Load Balanced Clusters in Wireless Sensor Networks

Mohammad Hammoudeh*, Saeed Abuzour*, Sarah Mount†, Omar Aldabbas‡, Mai Alfawair §

*Department of Computing, Manchester Metropolitan University, Manchester, UK, m.hammoudeh@mmu.ac.uk
†School of Computing and IT, University of Wolverhampton, Wolverhampton, UK, s.mount@wlv.ac.uk
‡Al-Balqa Applied University, Faculty of Engineering, Jordan, Al-Salt, o.aldabbas@bau.edu.jo
§Al-Zaytoonah Private University of Jordan, Faculty of Science and IT, Jordan, m.alfawair@alzaytoonah.edu.jo

Abstract—This paper proposes a clustering algorithm - Balanced Minimum Radius Clustering (BMRC) - for use in large scale, distributed Wireless Sensor Networks (WSN). Cluster balancing is an intractable problem to solve in a distributed manner, and distribution is important, by reason of both avoiding specialised node vulnerability and minimising message overhead. The BMRC algorithm described here distributes several of the cluster balancing functions to the cluster-heads. In proposing this algorithm, several tentative claims have been made for it, namely that it is suitable for arbitrary number of cluster heads; that it specifies a way to elect cluster heads and use them to create the local models; that it accomplishes optimal balanced clusters in distributed manner; that it is scalable and it uses the number-of-hops as a clustering parameter; that it is energy efficient. These claims were studied and verified by simulation.

Index Terms—Wireless Sensor Networks, Clustering, Load-Balancing.

I. INTRODUCTION

Routing has proved to be a key issue in the area of WSNs research. Based on the literature, e.g. [1], it is well-understood that network lifetime, scalability, robustness, and the performance of the WSNs applications often depends profoundly on efficient and reliable communication. However, it is difficult to attain both scalable and robust communication in WSNs. As clustering approaches are particularly tempting to large-scale high-density sensor network applications, clustering is sought as a solution to provide the requirements stated above. In WSNs, clustering is the process of logical grouping a set of nodes into disjoint and homogeneous groups, called clusters, based on a shared property such as nodes geographical location. Nodes within the same cluster are more closely related to one another than nodes assigned to different clusters. Clustering has become an increasingly important task in modern WSNs domains. It is an effective technique to achieve scalability, self-organisation, reduce control messages, power saving, bandwidth reusability, channel access, routing, enhanced resource allocation, and others. Therefore, clustering is another very important optimisation problem in WSNs. Many protocols use heuristics to elect cluster-heads. These heuristics are based on the minimisation of both transmission distances and the number of cluster heads.

Cluster-based routing is one of the most popular routing schemes in WSNs [2], [3], [4], [5], [6], [7], [8], [9]. It is a two or more tier routing scheme known for its scalability and communication efficiency. Nodes in the upper tier are called cluster heads and act as a routing backbone, while nodes in the lower tier perform the sensing tasks. A sink-based single tier network can lead to congestion at the gateway especially in dense sensor networks. This can cause communication delays and inadequate tracking of the sensed events. Moreover, some of the routing algorithms for such network architecture are commonly not scalable, e.g. [10]. To overcome these problems, network clustering has been proposed as a possible solution. In this paper a clustering algorithm that satisfies the following requirements is proposed:

1) Scalability: Clustering approaches are particularly tempting to large-scale high-density sensor network applications.

2) Load-balancing: One of the main challenges in WSNs systems design is balancing the resource usage of individual nodes while maintaining the desired global network behaviour. Clustering algorithms must be able to organise nodes in such a way that communication and processing load is minimised as well as energy consumption is distributed equally among nodes to achieve longer network life. Distributing workload amongst nodes will also help to prevent energy depletion in one part of the network.

3) Clustering: Clustering must be kept simple and decentralised. Each node should be able to independently make its decisions based on the available local information. A distributed implementation of clustering algorithms is expected to create minimal communication overload on the sensor nodes. Clusters setup should be efficient in terms of processing and communication. Furthermore, the clustering algorithm should assist achieving load-balancing requirements through fair distribution of sensor nodes among various clusters. Also, the total number of clusters in the sensor network should be sustained equal or around the optimal number of clusters defined by [11] which is 5% of the total number of nodes in the network.

In this paper we study a set of clustering algorithms in the literature, particularly the algorithm presented in [12]. Based on the findings in the literature we propose a new distributed clustering algorithm that solve some of the problems described
The following Section describes a number of clustering techniques for WSNs. Section III describes the BMRC clustering algorithm proposed by the authors and Section IV gives an experimental evaluation of the technique, carried out in simulation. Section V concludes.

II. RELATED WORK

In this section, we briefly review the main clustering protocols used in this paper, e.g. for comparison, and we refer the interested reader to [13], [14], [15], [16], [17] and references there in for a comprehensive survey of the recent clustering algorithms.

Low Energy Adaptive Clustering Hierarchy (LEACH) [11] is one of the most promising routing algorithms for sensor networks. However, LEACH has been based on a number of assumptions which in the authors’ opinion limit its effectiveness in a number of applications. LEACH is well-suited for applications where constant monitoring is needed and data collection occurs periodically to a centralised location. It increases network lifetime in two ways. First, the load is distributed to all nodes but not at the same time. Second, there is lossless aggregation of data by the cluster-heads. The protocol is powerful and simple since nodes do not require global knowledge or location information to create clusters. Despite the significant overall energy savings, however, the unrealistic assumptions made by the protocol raise a number of issues. These assumptions are listed in [18].

MuMHR [18] is an improvement over LEACH. MuMHR provides solutions to some of the limitations of LEACH. The main objective of this protocol is to provide substantially energy-efficient and robust communication. Similar to LEACH, MuMHR does not generate optimally balanced clusters. This algorithm is studied in details in subsection III-A.

Balanced Clustering algorithm proposed in [12] studies the theoretical aspects of the clustering problem in WSNs with application to energy optimisation. The algorithm considers the clustering problem with the energy expenditure as an imperative optimisation parameter. The authors define an optimal clustering algorithm such that each cluster is balanced and the total distance between sensor nodes in the same cluster is minimised. In [12], the maximum distance between any pair of nodes in a cluster $C$ is called the diameter of the cluster. The algorithm is based on the theorem which states that for any cluster $P$ with the maximum diameter $d$, there is a cluster $P'$ with maximum diameter $d'$ such that $P'$ is linearly separable and $d' \leq d$. Since the linearly separable principle does not always hold in balanced problems, this algorithm is not suitable for the optimal balanced clustering. Particularly, the definition of the metric diameter is not appropriate to WSNs clustering as it considers the distance between any pair of nodes in a cluster instead of the distance between the node and its cluster-head. For instance two nodes in a cluster could be independent and not related and thus the distance between them has no effect on the cluster performance. Another drawback of this algorithm is that every node should have knowledge about all other nodes in the cluster, this knowledge is difficult to acquire, and causes communication overhead. Moreover, this algorithm does not consider the transmission range of nodes when moving them from one cluster to another. Finally, using the diameter instead of the distance between the nodes and their cluster-heads could result in energy inefficient clustering as shown in the example in Figure 1.

III. BALANCED MINIMUM RADIUS CLUSTERING

Balanced Minimum Radius Clustering (BMRC) is a clustering algorithm that generates optimally balanced clusters based on unbalanced clusters. The distributed balanced clustering consists of four different steps: (1) Local clustering; (2) Determination of a local model; (3) Determination of a global model which is based on all local models; (4) Finally, updating of all local models.

In BMRC, the nodes are clustered locally, then the respective cluster-head extracts a suitable representative information about its cluster. These representatives are sent to the sink node, which combines all local representatives to generate a balancing plan. This approach is efficient, because the local clustering can be carried out quickly and independently from each other. Furthermore, it achieves lower transmission cost, as the number of transmitted representatives is much smaller than the cardinality of the complete data set. Based on the small number of representatives, the global cluster balancing can be done very efficiently.

The proposed distributed balanced-clustering algorithm is carried out on two different levels, i.e. the local level and the global. On the local level, all sites carry out clustering independently from each other using MuMHR algorithm. After having completed the local unbalanced clustering, a local model is determined. Our proposed local models consist of a set of representatives for each locally found cluster. Each representative is a concrete description for nodes residing on the corresponding local cluster. BMRC builds initial network clusters-based algorithm. The resulting clusters are then modified by BMRC to form a load balanced clusters that are energy efficient. Next the local model is transferred to sink node, where the local models are merged in order to form a global model. The global model is created by analysing
the local representatives. This analysis is similar to a new clustering phase with suitable global clustering parameters. A global cluster-identifier is assigned to each local representative. This resulting global balanced clustering is sent to all local sites that start modifying their clusters to implement the global model. This is a very difficult step as there might exist dependencies between nodes located on different sites which are not taken into consideration by the creation of the local models. In contrast to a central balanced clustering of the complete network, the central balancing of the local clusters can be carried out much faster.

In today’s WSNs, communication is orders of magnitude more expensive than local computation. The amount of energy needed to transmit a message to a destination at distance \(d\) from the source can be calculated by the following formula: 
\[
e = kd^c
\]
where \(k\) and \(c\) are constants for a specific wireless system [12]. Therefore, minimising the distance helps to reduce the communication overhead and the energy dissipation thereafter. On the other hand, balancing the clusters is needed for evenly distributing the load among all cluster-head to avoid energy depletion at one area of the network. Therefore, BMRC is designed to utilise a combination of two clustering parameters.

1) Radius: is the maximum distance from a node to its cluster-head
2) Number-of-hops: is the number of intermediate nodes between a node and its cluster-head

As communication is the most costly task in terms of energy, it must be used mostly carefully. To minimise the bridging distance between nodes and their respective cluster-head one needs to minimise the Radius clustering parameter. Furthermore, forwarding messages at intermediate nodes to the next nodes involves turning the node transmitters on which increases the total amount of energy needed to transmit a message from a source to destination. The number-of-hops metric were also chosen as a second clustering parameter to reduce the total amount of energy consumption involved in transmitting a message through multi-hops. We define diameter as a hybrid clustering parameter which is composed of both: Radius and the number-of-hops. Using the hybrid clustering parameters, diameter, the full BMRC algorithm is written as shown in Algorithm 1.

**Algorithm 1** Balanced minimum Radius k-clustering create local clusters using MuMHR.

1: if node is a cluster-head node then
2: calculate local model
3: send local model to sink
4: end if
5: if node is sink then
6: then calculate global optimal balanced clustering model
7: send global model to all cluster-heads
8: end if
9: when cluster-head nodes receive the global model then
10: implement the changes defined by the global model

BMRC has many advantages including: (1) it is suitable for arbitrary number of cluster-heads; (2) it specifies a way to elect cluster-heads and use them to create the local models; (3) it accomplishes optimal balanced clusters in distributed manner; (4) it is scalable and uses the number-of-hops as a clustering parameter; (5) it is energy efficient; (6) it does not require global network knowledge.

Figure 2 sketches a simple example that illustrates how BMRC works. The first step is to generate local clusters that are mostly unbalanced. These clusters are shown in Figure 2 (a) where Cluster 1 has a single node and Cluster 2 has three nodes. Next, the cluster-heads generate cluster representatives and send it to the sink. For example, the representatives for Cluster 1 is \((H_1, (3, 3,1,1))\) and for Cluster 2 is \((H_2, (9,9,3, [2,4]))\). The elements of the tuple are: the cluster-head ID; cluster-head location; number of member nodes; and a list of nodes. When processing the local representatives, using the diameter metric the sink determines a global model which includes moving node 4 from Cluster 2 to Cluster 1. Upon receiving the global clustering model, the cluster-heads starts implementing that model by moving node 4 to Cluster 1. The result is optimally balanced clusters as shown in Figure 2 (b).

In the next subsections we discuss BMRC clustering steps in details.

**A. Local clustering**

In this step, nodes are clustered locally using any multi-hop clustering algorithm. In this work we use MuMHR [18] routing algorithm but any other multi-hop clustering algorithm can be deployed. It is always desirable to use energy efficient clustering algorithms that produce cluster with node distribution as close as possible to the optimal clustering in order to make the balancing step simpler. Furthermore, the deployed local clustering algorithm is important because it determines the cluster-heads who generate the local cluster representatives.

MuMHR (Multi-hop, Multi-path, Hierarchal Routing) is a wireless sensor networks cluster-based routing algorithm. It is an improvement over LEACH [11]. It relaxes some of the unrealistic assumptions made by LEACH such as the single hop communication. The main objective of MuMHR protocol is to provide substantially energy-efficient and robust communication. The energy efficiency is achieved by load balancing at two levels: (1) at the network level, which involves traffic multiplexing over multiple paths; (2) at the cluster level, introducing rotation of the cluster-heads every given interval of time.
The operation of the proposed routing protocol can be split into two phases: the setup phase and the data transfer phase. During the setup phase, cluster-heads are selected and hierarchy is created. During the data transmission phase, sensor nodes transmit data to their cluster-head. The cluster-head aggregates the received data before transmission to the sink or immediately multiplex messages over multiple lines in time critical applications.

MuMHR reduces the energy expenditure by shortening the distance between the node and its cluster-head and by reducing the setup communication overhead. This is done through incorporating the number-of-hops metric together with the back-off waiting time. The back-off waiting time helps to decrease the number of set-up messages and aid the formation of more geographically uniform clusters. During the back-off waiting time, sensor nodes receive advertisement messages and only consider the message with the smallest number-of-hops received during that time.

Although MuMHR aims at load balancing, the results published in [18] shows that node distribution among clusters is not even. BMRC algorithm generates optimally balanced clusters based on these unbalanced clusters.

B. Determination of a local model

After having the nodes clustered locally, we need a small number of representatives which describe the local clustering result accurately. We have to find an optimum trade-off between the following two mutually conflicting requirements: cluster representatives should be compact as much as possible; and provide an accurate description of a local cluster. As the maximum cluster diameter and the list of cluster member nodes are computed during the clustering phase, it might serve as good representatives. Unfortunately, the number can become very high, especially in very dense networks. Therefore, we define a list, $L$, of nodes which contains all nodes that are $diameter/2$ far from the cluster-head. The local model also contains the cluster-head ID and location, number of nodes that are members of that particular cluster, and the list $L$. The cluster-head location is used by the sink to find adjacent clusters that possibly can exchange members to achieve optimal cluster balancing based on the received diameter parameters of local clusters.

C. Determination of the global model

To find a global clustering model, we use Algorithm 2. The aim is to create optimally balanced clustering where nodes are evenly distributed over different clusters using only the local model information available at the sink. The sink node will decide how the local clusters are going to be modified and disseminate this information to all cluster-heads. The sink decisions are a set of sensor node moving steps from one cluster to another based on the defined hybrid diameter parameter. Each cluster could exchange nodes with one or more adjacent clusters to arrive to an optimal energy efficient node distribution.

<table>
<thead>
<tr>
<th>Algorithm 2 Calculation of the global model at the sink</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Input: Set of models $S_n$</td>
</tr>
<tr>
<td>2: Output: $n$ disjoint clusters $S_1, ..., S_n$ with minimum $\max{diameter(S_1), ..., diameter(S_n)}$</td>
</tr>
<tr>
<td>3: Find the biggest cluster of two adjacent clusters $C_1$ and $C_2$</td>
</tr>
<tr>
<td>4: if $(</td>
</tr>
<tr>
<td>5: Swap $C_1$ and $C_2$</td>
</tr>
<tr>
<td>6: end if</td>
</tr>
<tr>
<td>7: while $(</td>
</tr>
<tr>
<td>8: Find a point $v \subset C_1$ such that $diameter(C_2 \cup {v} - diameter(C_2)$ is minimised</td>
</tr>
<tr>
<td>9: $C_1 \leftarrow C_1 \setminus {v}$</td>
</tr>
<tr>
<td>10: $C_2 \leftarrow C_2 \cup {v}$</td>
</tr>
<tr>
<td>11: end while</td>
</tr>
</tbody>
</table>

D. Updating of the Local Clustering based on the Global Model

After having created a global clustering model, the sink will transmit the complete global model to all cluster-heads. The cluster-heads collaborate to modify their cluster using the global model such that each cluster has equal number of nodes while minimising the diameter parameter.

IV. EXPERIMENTAL EVALUATION

In order to implement the BMRC algorithm and study its properties, a WNS simulator called Dingo [19] was used. Dingo is a development tool for WSNs. Dingo features a customisable energy module which makes it more flexible than the energy model used by NS-2 [20] and other simulators. Moreover, Dingo features a significant improvement in the simulation performance by giving the option to split the visualisation from the simulation. It forces thread switching to occur so that threads can not dominate the scheduler which makes the GUI more responsive.

Dingo provides tools for the simulation and deployment of high-level, Python code on real sensor networks. For example, Dingo-boom provides a two-way interface between MoteIV’s Boomerang class motes and Dingo. Dingo-top is another tool which is used to dump network topology data to a text file and generate a graphical representation of that topology. Furthermore, Dingo has several features in the form of plugins. These can be activated/deactivated on the plugin menu. Also, Dingo has a "Topology" menu which can be used to change the network topology of a simulation from a random topology to/from a grid. Network topologies can be loaded and saved.

For our experiments, we created a 50-node network, where the nodes are scattered randomly on $600 \times 600$ grid, such that no two nodes share the same location. The transmission range of each node is bound to $100m$. The processing delay for transmitting a message is randomly chosen between 0 and $5ms$, simulating real-world characteristics of low-power radio transmission.

We compare the performance of BMRC with that of MuMHR and the balanced minimum diameter algorithm.
Network configuration remains one of the most problematic tasks in WSN design. The largest practical networks so far have been 'designed' - that is, the hierarchy has been fixed and built in at design time. This approach has resulted in an inevitable vulnerability to poor surveying of the real environment of the network site and to subsequent equipment failures, which render the designed-in architecture non-viable. For this reason, self-configuration remains a vitally important area of research, and algorithms such BMRC are of great interest in finding a practical solution.

V. CONCLUSION

Figure 3 compares the distribution of nodes among clusters formed using BMRC with those formed using MuMHR. In MuMHR, it can be clearly seen that there is no optimal uniform distribution of node amongst the clusters, which increases both the heavy clusters management overhead and also the energy consumption. Whereas in BMRC, nodes were distributed much more fairly among clusters with a standard deviation of $\pm 1.6$. The standard deviation of MuMHR was approximate triple of that generated by BMRC. Figure 4 shows the clustering results of the balanced minimum diameter algorithm against those of the BMRC. The former algorithm achieved a standard deviation of approximately 2.9. These results demonstrate that the BMRC algorithm outperformed both MuMHR and the balanced minimum diameter algorithm. The BMRC algorithm generated clusters with almost equal number of nodes.

REFERENCES