IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON INTELLIGENT DATA SENSING, COLLECTION AND DISSEMINATION IN MOBILE COMPUTING

Received April 13, 2019, accepted May 5, 2019, date of publication May 9, 2019, date of current version May 22, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2915957

QSDN-WISE: A New QoS-Based Routing Protocol for Software-Defined Wireless Sensor Networks

XIAOBO TAN[1,4], HAI ZHAO[1], GUANGJIE HAN[2,3], (Senior Member, IEEE), WENBO ZHANG[4], AND TENG ZHU[4]

[1]School of Computer Science and Engineering, Northeastern University, Shenyang 110819, China
[2]School of Information Science and Technology, Qingdao University of Science and Technology, Qingdao 266061, China
[3]Department of Information and Communication System, Hohai University, Changzhou 213022, China
[4]School of Information Science and Engineering, Shenyang Ligong University, Shenyang 110159, China

Corresponding author: Guangjie Han (hanguangjie@gmail.com)

The work was supported in part by the National Key Research and Development Program under Grant YS2017YFGH001945, in part by the National Natural Science Foundation of China-Guangdong Joint Fund under Grant U1801264, in part by the Six Talent Peaks Project in Jiangsu Province under Grant XYDXXJS-007, in part by the General Project of Liaoning Education Department under Grant L2015465, in part by the Key Discipline Open Fund of Shenyang Ligong University, and in part by the Liaoning Natural Science Foundation under Grant 20170540793.

ABSTRACT Today, a wide variety of applications with different requirements are rapidly developed in industrial wireless sensor networks, and providing the Quality of Service (QoS) for this kind of communication network is inevitable. It is difficult to solve the problems of poor adaptability and difficulty in the implementation of the QoS-based network configuration and management in traditional network architecture. We present a hierarchical software-defined network architecture for wireless sensor networks, which makes the complex network management possible and the system more adaptable. Furthermore, we propose a QoS-based routing protocol, called QSDN-WISE, which consists of a clustering algorithm, a routing algorithm, and local network maintenance. A double-cluster head-based uneven clustering algorithm, called DCHUC, avoids the energy hole phenomenon and reduces the workload of a single cluster head. The centralized QSDN-WISE routing algorithm constructs two heterogeneous forwarding paths for nodes, which meets the requirements for different data levels. Local network maintenance reduces the number of control messages in the network. The simulation results indicate that the QSDN-WISE can provide the QoS support for data with different requirements, balance the network energy consumption, and prolong the network's lifetime.

INDEX TERMS Wireless sensor networks, QSDN-WISE, DCHUC, clustering algorithm, energy hole, routing algorithm, network lifetime.

I. INTRODUCTION

In recent years, with the rapid development of wireless sensor network technology, there have been increasingly more applications for wireless sensor networks (WSNs) [1]–[4]. Different application data have different requirements for QoS, including time delay and reliability, in various applications of industrial wireless sensor networks. Even within the same application, data of different types require sensor networks to operate, store, and transmit the data in different ways [5]. Therefore, designing a reliable and adaptable network communication protocol to meet QoS requirements for different applications in industrialized environments, and improving the overall performance of the network, has become an important research objective in the industrialized application of wireless sensor networks.

Researchers have proposed many improved QoS routing protocols based on early classical QoS routing protocols, such as SAR, SPEED, and EQR, and some have been applied in practice [6], [7]. Shortly after IETF proposed the RPL routing protocol, [8]–[10] adopted the data classification mechanism and designed independent objective functions for different types of data to meet QoS requirements for wireless sensor network applications that was based on it. However, the routing protocols mentioned above, which are based on the existing architecture, still face some bottlenecks in practical applications:

(1) Distributed routing protocols are usually adopted in large-scale deployment of WSNs, which can only choose an appropriate path based on local network

The associate editor coordinating the review of this manuscript and approving it for publication was Anfeng Liu.

information [11], [12]. However, it is difficult to choose the best path and achieve global optimization. Moreover, the large number of control messages generated in distributed routing protocols also has an impact on data forwarding.

(2) Many communication and computation resources are required in the process of establishing and maintaining routing for large-scale deployment of WSNs, which imposes great burdens on sensor nodes with limited energy, communication, and computation capabilities [13], [14].

(3) Energy, computation, and storage resources of sensor nodes are limited, while the routing protocols satisfying the diversity of QoS for WSNs are often very complex [15], [16]. Therefore, it is hard to achieve the configuration and management of protocols.

(4) Routing protocols are developed and solidified in sensor nodes by manufactures for specific applications of WSNs, and they are therefore hard to upgrade or replace [17], [18]. This results in poor adaptability and reusability.

Therefore, new technologies must be introduced into the applications of industrial WSNs to remedy the shortcomings mentioned above.

The Software Defined Network (SDN) originates from the Clean Slate project of Stanford University, and is a new network architecture that is one of the most promising solutions for the future Internet. In the SDN network, the logically centralized control plane is separated from the forwarding plane. The control plane manages and configures the network, and deploys new network protocols in a programmable manner through open and unified interfaces. Network equipment only forwards data according to the strategy sent from the controller. Via the centralized control of SDN, the control plane can acquire network information, such as network topology and traffic in time, so as to achieve more flexible and effective management than the traditional network architecture. The network equipment in SDN uses general hardware, so it is more convenient to deploy, maintain, and upgrade the network [19].

As soon as it was proposed, SDN attracted wide attentions from academia and business circles for its innovation and advantages, and it has achieved some theoretical research results and practical application cases in wired networks [20], [21]. After it proved successful in traditional wired networks, SDN was introduced into WSNs, Wireless Mesh Networks, and other wireless networks to improve the overall performance of the network and reduce the cost of network management [22]. After substantial experimentation, researchers have proven that centralized management based on Software Defined WSN (SD-WSN) has more advantages in the aspects of sensor node simplification, efficient network management and diagnosis, and network parameter configuration, among others, compared with the traditional distributed management of WSNs [23]. Furthermore, centralized management based on SD-WSN has been considered to better adapt to the future development of WSNs, and this view has been widely accepted by academia [24].

At present, the research on software-defined wireless sensor networks mainly focuses on the network management architectures. Research on routing protocol, especially QoS routing protocol based on SD-WSN, is still in the initial stage. In this paper, we propose a new QoS-based routing protocol for software-defined wireless sensor networks called QSDN-WISE that is based on SDN-WISE. The main contributions of the current study are summarized as follows:

(1) We develop a centralized architecture based on SDN-WISE, which makes the complex network management possible and the system more adaptable.

(2) We propose a double cluster head-based uneven clustering algorithm called DCHUC, which reduces the workload of cluster heads and avoids the energy hole problem.

(3) We present a QoS-based routing protocol based on SDN-WISE called QSDN-WISE, which can provide QoS support for data with different requirements.

The remainder of this paper is organized as follows. In Section 2, some related work on software-defined WSNs is examined. Section 3 explains the data classification and proposed SDN-based architecture for hierarchical WSNs. Section 4 presents QSDN-WISE in detail. The simulation and analysis are shown in Section 5. Section 6 concludes this paper.

II. RELATED WORK

Numerous recent studies have focused on software-defined wireless sensor networks. In 2012, Luo and Mahmud almost simultaneously proposed that SDN and WSN should be integrated to solve the inherent problems in traditional WSNs, and became the pioneers of Software Defined WSN research [25], [26]. Luo designed Sensor OpenFlow based on OpenFlow protocol to separate the control plane from the data plane of sensor nodes. Mahmud proposed a programmable sensor called Flow-Sensor and replaced the traditional WSNs with software-defined WSNs, thus constructing SD-WSN with programmability and customization. In 2015, the research on software-defined wireless sensor networks became a prevalent topic. Reference [27] uses standardized commercial hardware and SDN technology to design a repeatable configurable WSN framework, which discards the OpenFlow protocol, adopts loose-coupling method, and endows sensor nodes with control functions partly. Each sensor node in the framework executes policies through a microcontroller, and the parameters for the whole network are configured globally through a centralized controller. Reference [28] proposes a new state-based SDN-WISE framework, which is designed not only to reduce the amount of data exchanging between the controller and sensor, but also to make sensors support a state-based programmable mode by finite state machine. In this way, sensor nodes in SDN-WISE can run operations that are not supported in stateless solutions. In addition, SDN-WISE provides intact communication protocols and API interfaces, which provides a potential basis for its further research and development. The SDN-WISE protocol stack is illustrated in Fig.1. Reference [29] proposes

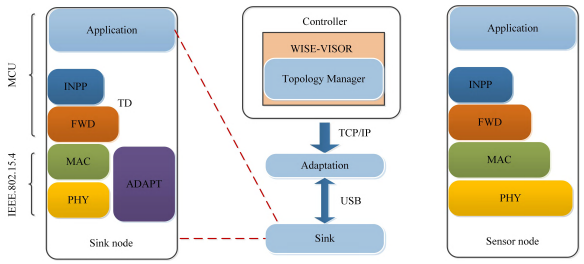


FIGURE 1. SDN-WISE protocol stack.

an implementation method for sensors using MCUs and FPGAs with ultra-low energy consumption to support SDN, which lays a foundation for the practical application of SD-WSN.

At present, a few researchers have focused on routing protocols based on SD-WSN architecture. Reference [30] proposes a QoS-based mechanism that is based on SDN-WISE. In the mechanism, the controller uses the state to give information about the congestion condition at the node, which can deal with traffic flows for different QoS requirements. Reference [31] proposes a disjoint multipath routing protocol based on SDN (SDN-DMRP). In SDN-DMRP, the controller collects relevant information about sensor nodes, calculates disjoint multipath forwarding routes for sensor nodes through a centralized algorithm, and sends them to sensor nodes in the form of flow tables. SDN-DMRP only uses hop count as parameter to choose the next hop, and does not provide QoS support for the network.

III. THE PROPOSED MODEL

A. DATA CLASSIFICATION

At present, sensor nodes can simultaneously collect event data, periodic data, and monitoring data of other types [32], [33]. Generally, after a sensor node collects event data, such as alarm data, it is necessary to ensure that the monitoring data is transmitted to the sink node in a timely and reliable manner, which ensures that the users can take measures timely and effectively [34]. Periodic monitoring data collected by sensor nodes, such as temperature and humidity, need periodic updates, so they do not require real-time and reliability. However, audio and video data collected by sensor nodes often need to ensure a certain real-time performance. For sensitive targets in the monitoring environment, batch monitoring data in a short time often needs to be acquired. However, such monitoring data often does not require low time delay, but requires certain reliability. It is difficult for sensor nodes to meet different QoS requirements for the data if they deal with data of different types in the same way. Therefore, a routing protocol supporting QoS needs to adopt a classification mechanism for monitoring data.

In order to meet the different requirements of real-time performance and reliability for different types of data, the QSDN-WISE routing protocol proposed in this paper classifies the monitoring data of different types according to their different sensitivities of time delay and packet loss rate. The result of data classification is provided in Table 1.

TABLE 1. Data classification table.

Priority	Time Delay	Packet Loss Rate	Specific Data Type
1	sensitive	sensitive	event data such as alarm data
2	sensitive	insensitive	audio and video data
3	insensitive	sensitive	batch data for sensitive targets
4	insensitive	insensitive	periodic data such as temperature and humidity

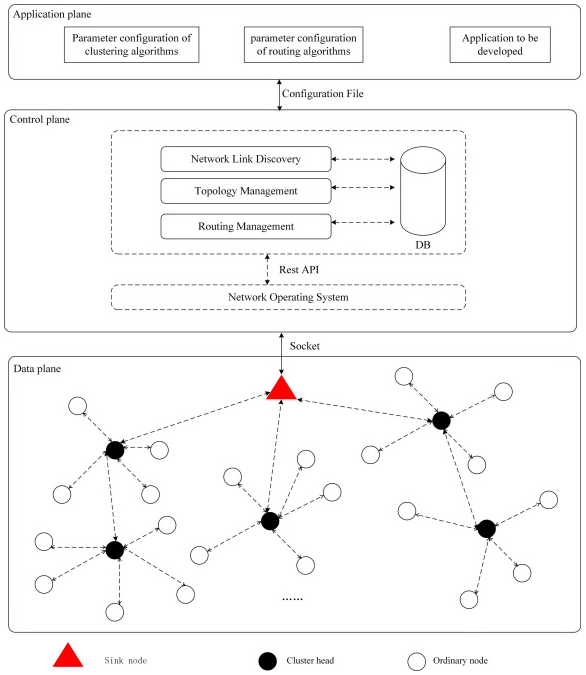


FIGURE 2. SDN-based Architecture for hierarchical WSNs.

B. SDN-BASED ARCHITECTURE FOR HIERARCHICAL WSNS

The proposed architecture is outline in Fig.2, and consists of three components: the application plane, the control plane, and the data plane. The application plane can dynamically configure the parameters of the clustering algorithm and routing algorithm through the configuration file. The control plane consists of a network link discovery module, a topology management module, and a routing management module. The data plane consists of a sink node and sensor nodes that support the QSDN-WISE protocol. The control plane interacts with the data plane through a socket.

In order to support the functions of SDN, the SDN-WISE client and flow table are added into the sensor nodes. In order to adapt to hierarchical network topology, sensor nodes in the data plane are divided into ordinary nodes and cluster-head

nodes according to their functions. Ordinary nodes only realize the function of data acquisition and send monitoring data to the cluster head, while cluster heads have the functions of data fusion, maintenance of cluster members, and data forwarding, in addition to monitoring the data acquisition function. The cluster information and maintenance information generated in the controller are only sent to the cluster heads. Therefore, the maintenance function of a cluster head is to receive cluster information and maintenance information from the controller, and transmit them to its member nodes in the manner of the flow table.

The control plane implements three main functions, including network link discovery, clustering-based topology management and maintenance, and routing construction and maintenance, based on backbone nodes.

1) NETWORK LINK DISCOVERY

The network link discovery module is responsible for discovering and maintaining the status information of sensor nodes in the network, including node ID, node neighbor table, node residual energy, node congestion, and link stability, which are provided for clustering-based topology management and routing construction and maintenance. Sensor nodes in the network use a classical Link Layer Discovery Protocol (LLDP), such as IEEE 802.15.4, 802.11b, etc., to implement the distributed link discovery algorithm, send information about the nodes and links to the controller, and then save them in the database.

2) TOPOLOGY MANAGEMENT AND MAINTENANCE

Clustering is an effective method for the management of large-scale deployment of wireless sensor networks. Therefore, the topology management and maintenance module utilizes the related status information of the nodes and links obtained by the network link discovery module to execute the centralized clustering algorithm and cluster maintenance, and saves the results to the database.

3) ROUTING MANAGEMENT

The routing management module conducts a centralized routing algorithm to construct and maintain QoS routing for the backbone network by utilizing the related status information of the nodes and links, clustering result information, etc., and saves the results to the database.

IV. QSDN-WISE ROUTING PROTOCOL

The QSDN-WISE routing protocol proposed in this paper includes four parts: the clustering algorithm, routing algorithm, cluster maintenance, and routing maintenance. During the initialization process, the controller conducts clustering and routing algorithms to complete initial network construction. Then the controller realizes the event-triggered cluster and local routing maintenance based on the global topology information, which is updated regularly.

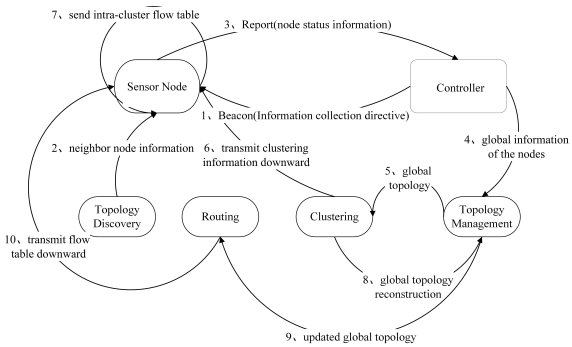


FIGURE 3. Process of clustering & routing algorithm.

A. NETWORK INITIALIZATION PROCESS

The network initialization process of QSDN-WISE is illustrated in Fig.3. The controller broadcasts a beacon control message to the whole network through the sink node, and initiates the distributed topology discovery process (TD) to obtain effective information of sensor nodes, such as node congestion, link stability, node residual energy, node neighborhood information, node ID, and RSSI. The data flows from points 1 to 4 complete the process of network topology discovery and form the initialization network topology. After completing the topology discovery process, the topology management module in the controller conducts the DCHUC clustering algorithm to cluster the network based on the global topology information of the network. The data flows from points 5 to 8 complete the clustering process. After the network clustering is completed, the routing management module constructs a backbone routing tree supporting QoS for the backbone network. The data flows from points 9 to 10 complete the routing process.

B. DCHUC CLUSTERING ALGORITHM

DCHUC is a centralized algorithm executed in the controller, so it not only can realize global optimization, but can also reduce the number of control messages generated in the network, which further reduces the frequency of network storm and network energy consumption. Furthermore, DCHUC adopts a non-uniform clustering mechanism, so it ensures the formation of more clusters in the area near the sink to forward network data, which avoids the energy hole problem and prolongs the network lifetime. In addition, DCHUC adopts the double cluster head mechanism, in which double cluster heads are elected with low node congestion and high link stability, respectively, according to the factors of node congestion, link stability, and node residual energy. This not only reduces the workload of single cluster head, but also supports QoS services intra-cluster. Although the double cluster head mechanism of DCHUC forms two different network topologies, which increases the difficulty in management and control of the network, the centralized management mode based on SD-WSN architecture makes up for this defect.

DCHUC includes two processes: the election of cluster head and the clustering of ordinary nodes.

1) ELECTION OF CLUSTER HEAD

In order to achieve non-uniform clustering, DCHUC applies the Min-Max Scaling idea and calculates the competition radius for each sensor node with eq. (1).

$$r_{cmp}(n_i) = (1 - c \times \frac{d_{max} - d(n_i)}{d_{max} - d_{min}}) \times r_{max} \quad (1)$$

In eq. (1), $r_{cmp}(n_i)$ denotes the competition radius of node n_i; r_{max} is a constant number which represents the preset maximum competition radius; d_{max} and d_{min} denote the maximum and minimum distance of all nodes from the sink node; $d(n_i)$ denotes the distance between node n_i and the sink node; finally, c is an adjusting parameter to ensure that the competition radius is more reasonable.

The QSDN-WISE proposed in this paper requires ordinary nodes in a cluster to send data to the cluster head directly. According to the energy consumption model [35], [36], it has been concluded that the competition radius of clusters should be less than d_0 as much as possible in order to save network energy consumption. Therefore, we set the competition radius of clusters (r_{max}) to $r_{max} \leq d_0$. d_0 is a constant value of about 87 m. Eq. (1) ensures that there are more cluster heads near the sink node.

DCHUC considers the factors of node congestion, link stability, and node residual energy to elect the cluster head. The controller calculates the competition radius for each node according to eq. (1), and maintains a neighbor list ($L_{neighbor}$) for each node to store the ID of the neighbor nodes that are within the competition radius of the node. Then, the controller calculates the probabilities of becoming a low node congestion cluster head and a high link stability cluster head for each node according to eqs. (2) and (3), respectively.

$$P_{NC}(n_i) = \frac{E_{residual}(n_i)}{\bar{E}_{neighbor}(n_i)} \times (1 - NC(n_i)) \quad (2)$$

$$P_{LS}(n_i) = \frac{E_{residual}(n_i)}{\bar{E}_{neighbor}(n_i)} \times LS(n_i) \quad (3)$$

In eq. (2), $P_{NC}(n_i)$ denotes that node n_i becomes the cluster head with a low node congestion degree. $E_{residual}(n_i)$ denotes the residual energy of the node n_i. $\bar{E}_{neighbor}(n_i)$ denotes the average residual energy of the node n_i. $NC(n_i)$ denotes the congestion degree of the node n_i. In eq. (3), $P_{LS}(n_i)$ denotes the link stability of the node n_i. Node congestion and link stability are shown in eq. (4) and eq. (5) [37].

$$NC(n_i) = \frac{Q(n_i)}{L(n_i)} \quad (4)$$

$$LS(n_i) = \frac{S(n_i)}{M(n_i)} \quad (5)$$

In eq. (4), $Q(n_i)$ denotes the buffer queue length of node n_i which has been occupied. $L(n_i)$ denotes the total buffer queue length of node n_i. In eq. (5), $S(n_i)$ denotes the number of packets sent by the node n_i that arrives to its neighbor nodes successfully. $M(n_i)$ denotes the number of packets sent by the node n_i to its neighbor nodes.

TABLE 2. Pseudo-code of cluster head election.

Centralized Cluster Head Election
Initialization: there is global network topology information in the controller 1.Calculate($L_{nodes}, P_{NC}, P_{LS}$) 2.SORT($L_nodes, P_{NC}, P_{LS}$) 3. $(LP_{NC}, LP_{LS}) \leftarrow L_{nodes}$ 4.while $LP_{NC}.len()! = 0$ do 5. $j = 0$ 6. $i \leftarrow LP_{NC}[j]$ 7. for $LP_{NC} : LCH_{NC}.add(i)$ or for $LP_{LS} : LCH_{LS}.add(i)$ 8. for n in $L_{neighbor}(i)$ do 9. $LP_{NC}.delete(n)$ 10. for $LP_{LS} : n.LCCH_{NC}.add(i)$ or for $LP_{LS} : n.LCCH_{LS}.add(i)$ 11. end for 12. $LP_{NC}.delete(i)$ 13.end while 14.Repeat from 4 to 11 for LP_{LS}

After calculating the probability of becoming a low node congestion cluster head and high link stability cluster head, respectively, for all the nodes, the controller sorts all the nodes by node congestion in ascending order and by link stability in descending order, respectively, and saves the corresponding *IDs* of the nodes to the node congestion cluster head probability list (LP_{NC}) and the link stability cluster head probability list (LP_{LS}), respectively. Then, the controller successively stores the *IDs* in (LP_{NC}) and (LP_{LS}) o the cluster list with low node congestion (LCH_{NC}) and the cluster list with high link stability (LCH_{LS}), respectively, and deletes the node ID and its neighbor node *IDs* from (LP_{NC}) and (LP_{LS}) until (LP_{NC}) and (LP_{LS}) are empty, i.e. all nodes are covered by cluster heads. The pseudo-code for the centralized election cluster head process is presented in Table 2.

After cluster head election, two heterogeneous sets of cluster heads, each of which covers the whole network, are obtained: a low node congestion cluster head set (LCH_{NC}) and high link stability cluster head set (LCH_{LS}). In order to implement clustering for ordinary nodes, in the process of cluster head election, the controller maintains two candidate cluster head lists for each ordinary node, which are the low node congestion candidate cluster head list $LCCH_{NC}$ and the high link stability candidate cluster head list $LCCH_{LS}$.

2) CLUSTERING OF ORDINARY NODES

After cluster heads are elected, DCHUC goes into the clustering process of ordinary nodes. During the election of two types of heterogeneous cluster heads according to eqs. (2) and (3), a phenomenon in which one node becomes the cluster head of two types of heterogeneous clusters at the same time may occur, and it is allowed in DCHUC. However, in order to avoid data transmission between two heterogeneous cluster heads in one cluster, DCHUC does not allow a cluster head of one type to become a member node of the other type.

TABLE 3. Pseudo-code of clustering.

Centralized clustering algorithm
Initialization: there is global network topology information in the controller
1.for n in $L_{non-cluster}$ do
2. if $n.LCCH_{NC}.len() == 1$ then
3. $n.CH_{NC} \leftarrow LCCH_{NC}[0]$
4. $LCCH_{NC}[0].LCM_{NC}.add(n)$
5. else
6. $e \leftarrow MaxInClusterStrength(n.LCCH_{NC})$
7. $n.CH_{NC} \leftarrow e$
8. $e.LCM_{NC}.add(n)$
9. end if
10.end for
11.Repeat for $LCCH_{LS}$ and LCM_{LS} from 1 to 10

In the clustering process of a certain type of cluster, if there is only one cluster head in the competition radius of an ordinary node, this ordinary node will directly become a member of the cluster in which the cluster head is located. However, there may be multiple cluster heads within the competition radius of an ordinary node. In order to solve this problem, this paper defines clustering strength as shown in eq. (6). DCHUC calculates the clustering strength of each cluster in its competition radius for the node in this situation, and chooses to join the cluster with the highest clustering strength.

$$S_{incluster} = \theta E_{residual}(n_j) + \frac{1-\theta}{d_{n_i-n_j}}, \quad \theta \in (0,1) \qquad (6)$$

In eq. (6), $E_{residual}(n_j)$ denotes the residual energy of the cluster head n_j; $d_{n_i-n_j}$ denotes the distance between the node n_i at the intersection of multiple clusters and the cluster head n_j; finally, θ is a weight parameter with a value between 0 and 1. Eq. (6) shows that if the residual energy of cluster head n_j is higher, and the distance between the cluster head n_j and the ordinary node n_i is smaller, the ordinary node n_i will be more likely to become a member of the cluster in which n_j is located. Pseudo-code of clustering is presented in Table 3.

After the clustering process, the controller maintains two heterogeneous cluster head lists which are LCH_{NC} and LCH_{LS}, two cluster member lists for each cluster head which are LCM_{NC} and LCM_{LS}, and two cluster heads for each ordinary node which are CH_{NC} and CH_{LS}. After DCHUC, the controller generates a Cluster Head Notification message (CH_NP), a Cluster Member Notification message (CM_NP), and corresponding flow table rules, and transmits them to the corresponding sensor nodes. Fig. 4 illustrates the network topology generated by DCHUC.

As shown in Fig. 4, there are two types of cluster heads in the network after the DCHUC clustering algorithm, namely the low node congestion cluster head and the high link stability cluster head, and these two types of cluster heads form two hierarchical network topologies with their member nodes, respectively.

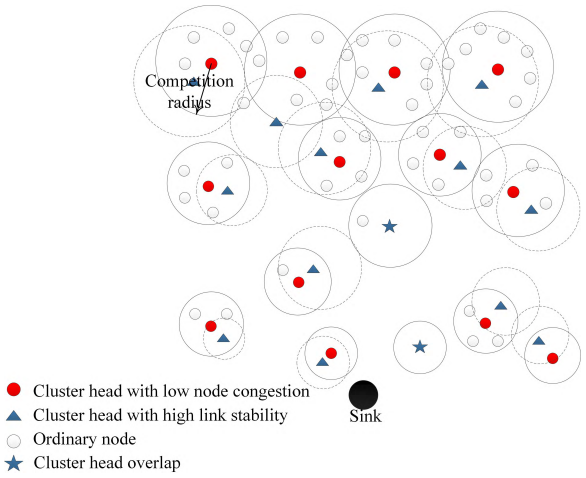


FIGURE 4. Non-uniform clustering map of DCHUC.

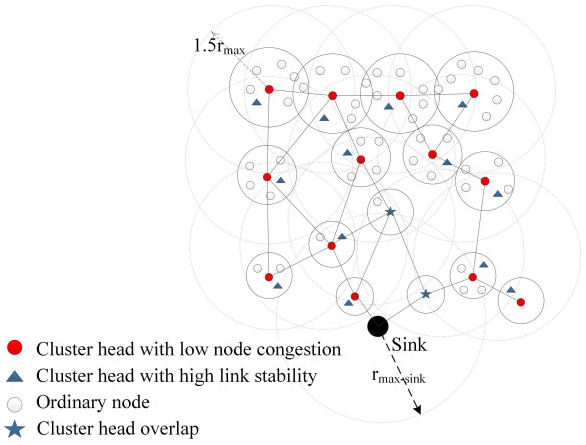


FIGURE 5. Undirected connection graph formed by cluster heads.

C. QSDN-WISE ROUTING ALGORITHM

In the QSDN-WISE routing algorithm, the controller adopts the same QoS routing algorithm for each hierarchical topology generated by DCHUC. Therefore, we will introduce the QSDN-WISE by using the backbone network composed of low congestion cluster heads as an example in this section.

The controller establishes an undirected connected graph for the backbone network composed of cluster heads and a sink node. In order to ensure the connectivity of the network, we set the communication radius of sensor nodes to be 1.5 times that of the competition radius, and established an undirected connected graph, as illustrated in Fig. 5. The sink node acquires its first hop of downward data transmission via the communication radius $r_{max-Sink}$ to further alleviate the energy hole. The value of $r_{max-Sink}$ is much larger than the communication radius of an ordinary sensor node, as the resources of the sink node are not limited.

After the undirected connected graph is formed, the controller chooses the best next hop for each node according

TABLE 4. Weight values for different data levels.

Data level	Time latency	Packet loss	α_k	β_k
1	sensitive	sensitive	1	1
2	sensitive	insensitive	1	0
3	insensitive	sensitive	0	1
4	insensitive	insensitive	0	0

to eq. (7).

$$OF_k(n_i) = \frac{\alpha_k \times NC(n_j) + \beta_k \times (1 - LS(n_j))}{E_{residual}(n_j)}, \quad (d_{n_j-Sink} < d_{n_i-Sink}) \quad (7)$$

In eq. (7), $E_{residual}(n_j)$ indicates that the node with high residual energy should be selected as its next hop; $NC(n_j)$ indicates that the node with low node congestion should be selected as its next hop if the data sent by the node requires low latency; $LS(n_j)$ indicates that the node with high link stability should be selected as its next hop if the data sent by the node requires high reliability; $d_{n_j-Sink} < d_{n_i-Sink}$ indicates that the node that should be selected as its next hop is the node with a distance from the sink that is less than the distance between n_i and the sink. Finally, α_k and β_k denote the weight parameters of the k^{th} data type, and their values range from [0,1]. We can conclude that the value of α_k should be greater than that of β_k for an important or emergency data type, while the value of α_k should be less than that of β_k for a non-emergency data type. In this paper, we set the weight parameters as follows: the value of α_k is set to 1 for the data type with real-time requirement, otherwise it is set to 0; the value of β_k is set to 1 for the data type with high reliability requirement, otherwise it is set to 0. Table 4 lists the corresponding values of α_k and β_k for the 4 data levels that are adopted in this paper.

The controller uses the objective function (OF) to choose the best next hop for each type of data of each backbone node. Routing algorithms supporting QoS for WSN usually adopt different OF formulas for different types of data, which increases the complexity of the routing algorithm and parameter configuration. In contrast, QSDN-WISE uses a unified OF formula and meets the QoS requirements of different types of data by only adjusting the two weight parameters of α_k and β_k, which reduces the complexity of the routing algorithm.

QSDN-WISE satisfies different QoS requirements for different types of data by setting different values of α_k and β_k. In Table 4, the objective function of $\alpha_k = 1$ and $\beta_k = 0$ satisfies the low-latency QoS requirement of data level 2, i.e. choosing the node with high residual energy and low node congestion degree as the next hop. If $\alpha_k = 0$ and $\beta_k = 0$, it indicates that QSDN-WISE chooses the next hop while only considering the node residual energy factor, which satisfies the QoS requirement of the data level 4 without requirements for time delay and packet loss rate. Therefore, the controller constructs an independent upward routing tree

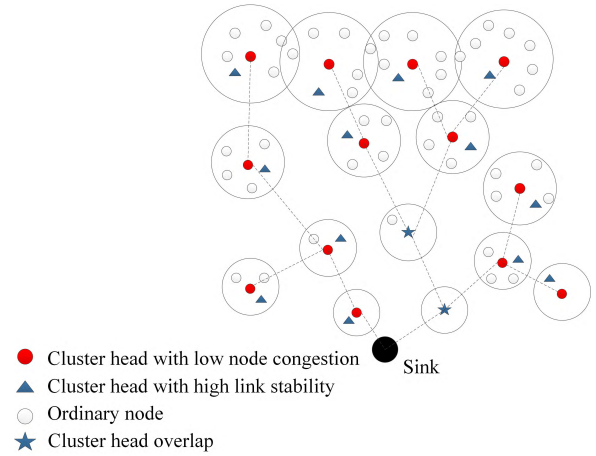


FIGURE 6. Upward routing tree for data level 2.

from the sensor node to the sink node by setting α_k and β_k for different types of data. The control information sent from the sink node to the sensor nodes requires high real-time capability and reliability, so the downward routing from the sink node to the sensor node is based on the routing tree of data level 1, and the reverse routing tree is established. Considering the high energy consumption of sensor nodes in the area near the sink node due to the requirement of forwarding much more data, the communication radius of the sink node is increased and the best next hop of the sink node in one hop range is chosen by eq. (7), so as to complete the reverse routing. Fig. 6 shows the final upward routing tree for data level 2.

D. MAINTENANCE FOR CLUSTERS AND ROUTING

Distributed clustering and routing algorithms in traditional network architecture often adopt periodic cluster head rotation and backbone network reconstruction. This process results in the periodic generation of a large number of control messages, which not only consumes too much node energy, but also interferes with the transmission of monitoring data. Although the IRPL routing protocol adopts event-triggered cluster head rotation and a local routing maintenance mechanism, which reduces the number of periodically controlled messages, a large number of control messages will still be generated in the network when the event is triggered. QSDN-WISE adopts event-triggered cluster head rotation and a local routing maintenance mechanism, and implements a centralized algorithm at the controller side, which avoids the generation of a large number of control messages when the event is triggered, thus improving network performance.

After the network initialization process of QSDN-WISE is completed, sensor nodes in the network regularly send state information, such as node residual energy, node congestion, and link stability, to the controller so that the controller can keep up-to-date global network information. Because of the small amount of data of node status information and the absence of a broadcasting mode, the network storm

phenomenon that is easily produced by a traditional distributed algorithm is avoided. The implementation of event-triggered cluster head rotation and local routing maintenance mechanism by the controller is described below.

1) CLUSTER HEAD ROTATION BASED ON EVENT TRIGGER

If the residual energy and the congestion degree of the cluster head is greater than the preset threshold, the link stability is less than the preset threshold, or the information of the cluster head cannot be updated by the controller, the cluster head rotation in the controller side will be triggered. The controller chooses eq. (2) or (3), according to the type of cluster head (low node congestion or high link stability, respectively), to calculate the probability of becoming the cluster head of this type, and chooses the node with the largest probability value as the successor cluster head. After the successor cluster head is elected, the controller generates the cluster head notification message (*CH_NP*), cluster member notification message (*CM_NP*), and the corresponding flow table rules, and sends them to the sensor nodes of the cluster.

2) LOCAL ROUTING MAINTENANCE

Event-triggered cluster head rotation can lead to the breakage of the global backbone network routing tree maintained by the controller. Therefore, after the controller performs the cluster head rotation, the backbone network needs to be partially repaired. First, the controller obtains the type of the original cluster head to determine which routing tree to repair. Then, the controller obtains all the child nodes in the routing tree, chooses the best next hop for the node and all its child nodes at each data level, and sends the corresponding flow table rules to the corresponding nodes. The process of choosing the best next hop has been described in detail in Section C.

E. QSDN-WISE PROTOCOL DESIGN

The QSDN-WISE proposed in this paper is an improvement of the SDN-WISE protocol, on which it is based. We will introduce the QSDN-WISE protocol in the following aspects: protocol stack, protocol message, and flow table structure.

1) QSDN-WISE PROTOCOL STACK

The QSDN-WISE protocol stack only modifies SDN-WISE on the controller side. The topology Manager (TM) of the SDN-WISE controller only generates planar network topology, which is difficult to adapt to WSN applications with large-scale deployment of sensor nodes. DCHUC cluster management is added to the TM module of the controller to generate a hierarchical network topology structure based on the SDN-WISE protocol stack. In order to support QoS routing, the controller of QSDN-WISE protocol stack adds a routing management module.

2) QSDN-WISE PROTOCOL MESSAGE

The header of SDN-WISE is modified as shown in Fig. 7. In order to meet the QoS requirements for different types of

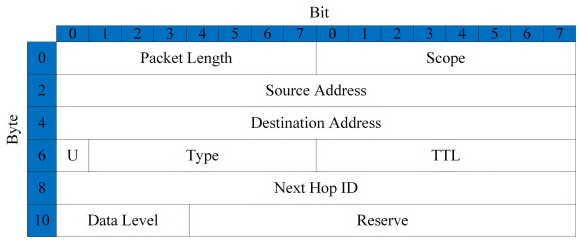


FIGURE 7. QSDN-WISE packet header format.

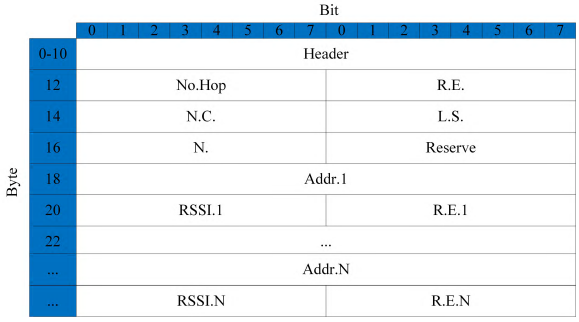


FIGURE 8. QSDN-WISE report packet format.

data, the data level field that occupies 4 bytes is added to the original SDN-WISE header to support up to 16 data levels.

The report message format of SDN-WISE is modified as presented in Fig. 8. Bytes from 0 to10 are the message header of the QSDN-WISE. The 12^{th} and 13^{th} bytes are the hop and node residual energy. The 14^{th} and 15^{th} bytes are the node congestion degree and link stability degree. The 16^{th} byte is the number of neighbor nodes. Starting from the 18^{th} byte, the node address, RSSI, and node residual energy of the neighbor nodes are defined.

In the QSDN-WISE protocol, the cluster head notification message (*CH_NP*) and cluster member notification message (*CM_NP*) are added based on the original SDN-WISE message.

a: CH_NP MESSAGE

The controller builds the *CH_NP* message according to clustering result and notifies the corresponding node to become the cluster head node. The format of the *CH_NP* message is exhibited in Fig. 9. Bytes from 0 to 10 are the message header of the QSDN-WISE. Starting from the 12^{th} byte, the ID and address information of all the member nodes of the cluster are defined. The value of the Type field in the *CH_NP* header is 8.

b: CM_NP MESSAGE

The controller builds the *CM_NP* message according to the clustering result and notifies the corresponding member nodes. The format of the *CM_NP* message is shown in Fig. 10. Bytes from 0 to 10 are the message header of the QSDN-WISE. The 12^{th} and 16^{th} bytes are IDs of the two cluster heads of different types. The 13^{th} and 17^{th} bytes

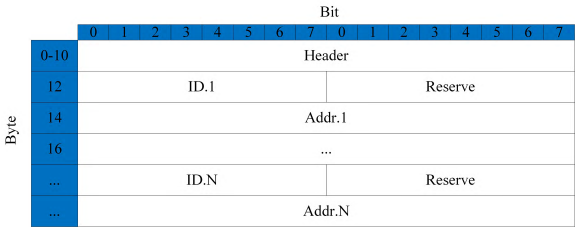


FIGURE 9. QSDN-WISE *CH_NP* packet format.

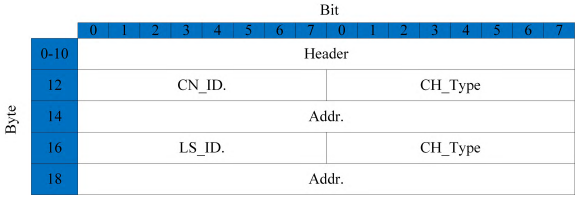


FIGURE 10. QSDN-WISE *CM_NP* packet format.

Matching Rule	Action
Packet[Dest. Addr.]==A and Packet[Data Level]==0	Forward(C)
Packet[Dest. Addr.]==A and Packet[Data Level]==1	Forward(D)

FIGURE 11. QSDN-WISE flow table example.

are the type of the cluster heads. 0 is the cluster head with low data congestion, and 1 is the cluster head with high link stability. The 14^{th}, 15^{th}, 18^{th}, and 19^{th} bytes are the addresses of two types of cluster heads.

c: QSDN-WISE FLOW TABLE

The QSDN-WISE protocol adds a data level field to the matching rules based on the original SDN-WISE flow table structure to match different data levels. Fig. 11 provides an example of the modified QSDN-WISE matching rules and behaviors. The destination addresses of the two flow table entries are the same in the figure, but the matching data levels are different and correspond to level 1 and level 2, respectively. The matched actions are also different. The data of matching level 1 will be forwarded to C, while the data of matching level 2 will be forwarded to D.

V. SIMULATION & ANALYSIS

A. SIMULATION SETTINGS AND PERFORMANCE INDICATORS

QSDN-WISE protocol is an improved protocol that is based on SDN-WISE, while the SDN-WISE project is still undergoing continuous improvement. At present, the JAVA version of SDN-WISE has become an open source project and has been published in GitHub [38]. The C version of SDN-WISE has been used in the experimental environment of real sensors. The QSDN-WISE protocol proposed in this paper is improved and simulated based on the JAVA version. The specific information of the simulation platform is provided in Table 5.

TABLE 5. Simulation platform information table.

Simulation development tool	Version information	Description
OS	Windows 7 64bit	Host operation system
Virtual machine	VMware Workstation 14 Player	Virtual environment
VmOS	Linux-3.13.0-62-generic	Operation system in VM
Sensor node simulation tool	Cooja	Contiki 3.0 simulation software
JDK	Openjdk-1.8.0_162	Java runtime environment version number
sdn-wise-api.jar	3.0.6	SDN-WISE development kit
Myeclipse	2016	IDE development environment

TABLE 6. Simulation parameter table.

Simulation parameter	Value
Network range	$(0,0)-(400,400)m$
Number of nodes	$[11-201]$
Node working state	$6.8mC/s$
Node energy consumption for sending packets	$0.0000027mC/byte/m^2$
Initial node energy	$5000mC$
Node voltage	$3.0V$
Buffer size of sensor node	120 data packets
d_0	$87m$

QSDN-WISE adds some parameters based on the simulation parameters provided by SDN-WISE, such as node buffer size and d_0. Table 6 describes the specific parameters.

B. SIMULATION RESULTS ANALYSIS

We compare QSDN-WISE with SDN-DMRP and SDN-WISE, which are based on SDN architecture, and IRPL, HEED, and EEUC, which are based on the traditional network architecture.

Fig. 12 shows the effect of adjusting the parameters c and r_{max} on clustering results. In Fig. 12, the abscissa is the value of r_{max} and the ordinate is the number of cluster heads after clustering. Each curve represents the variation of the number of cluster heads with r_{max} under the condition that c is a constant. It can be concluded that the number of cluster heads decreases with the increase of r_{max} if the adjustment parameter c is constant, while the number of cluster heads increases with the increase of the adjustment parameter c if r_{max} is constant. It can also be concluded from the simulation

Matching Rule	Action
Packet[Dest. Addr.]==A and Packet[Data Level]==0	Forward(C)
Packet[Dest. Addr.]==A and Packet[Data Level]==1	Forward(D)

FIGURE 12. DCHUC clustering result.

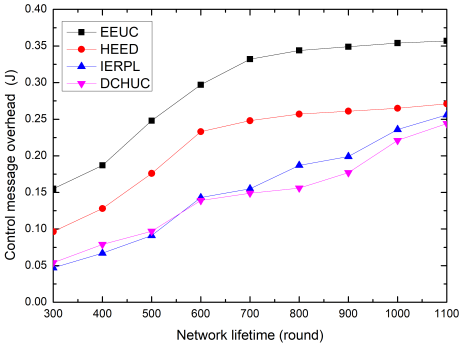


FIGURE 13. Comparisons of control message overhead.

result in Fig. 12 that there is no obvious jitter in the number of cluster heads, which indicates that the cluster heads are distributed evenly in the network.

Fig. 13 shows the comparison of the control message overhead between the DCHUC, HEED, EEUC, and IRPL clustering algorithms under the condition that the number of network nodes is 200. HEED and EEUC clustering algorithms perform the clustering algorithm periodically, so we count message overhead in rounds.

It can be concluded that the control message overhead of the above four clustering algorithms increases with time, and the control message overhead of DCHUC and IRPL is significantly less than that of HEED and EEUC. HEED and EEUC periodically execute their respective clustering algorithms in a round-based manner, resulting in a large number of control messages, while both DCHUC and IRPL adopt an event-based clustering algorithm. Large numbers of control messages are mainly generated in the initial stage of the network. After that, local topology maintenance is adopted, and the network generates fewer control messages. HEED chooses cluster heads randomly, which reduces the number of generated control messages. EEUC needs to exchange a large number of control messages among nodes when choosing cluster heads. Therefore, it is evident that the control message overhead of HEED is lower than that of EEUC. The control message overhead of DCHUC before 557 rounds in Fig. 13 is slightly greater than that of the IRPL clustering algorithm because the controller that implements centralized DCHUC after network initialization needs to periodically obtain the status information of all nodes in the whole network, and consumes a certain amount of message overhead. In contrast, IRPL only needs more control messages for topology maintenance, and there are no large-scale cluster head rotations before 557 rounds; therefore, the overhead is less than that of DCHUC. Starting from about 558 rounds, the control message growth of the IRPL clustering algorithm is greater than

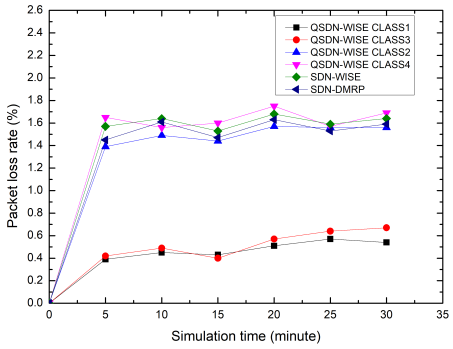


FIGURE 14. Comparison of packet loss rate.

that of DCHUC. With the increase of time, cluster head rotations are more frequent in IRPL, which requires more message overhead than that in DEHUC. In the simulation, we can also determine that the growth rate of the control messages of EEUC and HEED is extremely slow, while the growth rate of the control messages of IRPL and DCHUC increases significantly after the network operates about 700 rounds in EEUC, and about 600 rounds in HEED. Starting from about 600 rounds, more dead nodes gradually appear in the network running HEED and EEUC, which results in the reduction of the number of control messages. In contrast, cluster head rotations in IRPL and DCHUC become frequent, which significantly increases the number of control messages.

Fig. 14 illustrates the comparison of packet loss rate between QSDN-WISE, SDN-WISE, and SDN-DMRP protocols under the condition that the number of nodes is 200. In the simulation, the data are divided into four levels according to Table 4, namely CLASS1, CLASS2, CLASS3, and CLASS4. It can be concluded that the packet loss rates of CLASS1 and CLASS3 in the network running the QSDN-WISE protocol are significantly less than those of CLASS2 and CLASS4, because the data sensitive to packet loss rate in CLASS1 and CLASS3 will choose the node with high link stability as its next hop, thus reducing the packet loss rate of data forwarding. However, SDN-WISE and SDN-DMRP only adopt the hop number to choose its next hop, and do not support QoS service. Therefore, the packet loss rates in SDN-WISE and SDN-DMRP are close to the packet loss rates of CLASS3 and CLASS4 data. It is evident that QSDN-WISE provides QoS support for data sensitive to packet loss rate.

Fig. 15 displays the end-to-end delay comparison between QSDN-WISE, SDN-WISE, and SDN-DMRP protocols under the condition that the number of network nodes is 200. It can be concluded that the end-to-end delays of CLASS1 and CLASS2 in QSDN-WISE are significantly less than those of CLASS3 and CLASS4 data. The end-to-end delay in SDN-WISE is greater than that of CLASS1 and CLASS2 in QSDN-WISE, but less than that of CLASS3 and CLASS4 in QSDN-WISE. Although SDN-WISE considers the shortest path based on hop number, the cluster head must

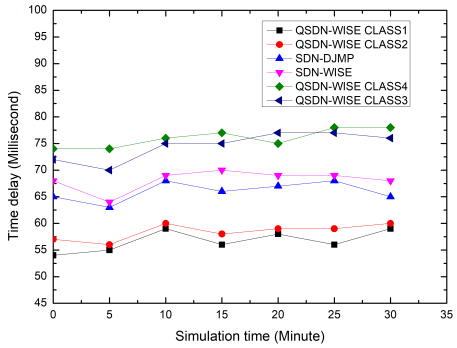


FIGURE 15. Comparison of end-to-end delay.

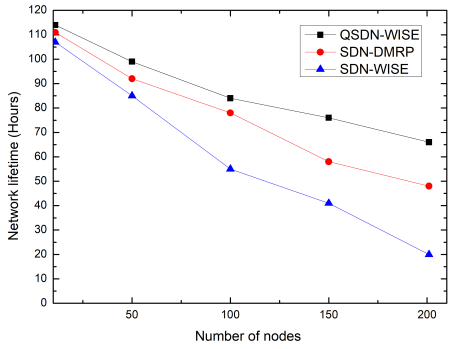


FIGURE 16. Comparison of network lifetime.

forward data of all levels in the cluster to the same relay node. When the relay node forwards data, the processing delay of the node will increase, and the processing delay accumulated during multi-hop forwarding will increase significantly. SDN-DMRP adopts multi-path routing based on the shortest path, so the time delay is less than that in SDN-WISE, but still greater than that of CLASS1 and CLASS2 in QSDN-WISE. The member nodes in the cluster in QSDN-WISE will forward data with low time delay to the cluster head with low node congestion degree, and the cluster head of this type will still choose the node with low node congestion degree as its relay node. Therefore, compared with SDN-WISE and SDN-DMRP, QSDN-WISE provides better QoS support for the data that needs low time delay.

Network lifetime is an important parameter for measuring the performance of wireless sensor networks. In the simulation, each node sends data packets at the same rate, and the simulation ends when the first node in the network runs out of energy. Fig. 16 shows the comparison between the network lifetime in different numbers of nodes in the network in the QSDN-WISE, SDN-WISE, and SDN-DMRP protocols. It can be concluded that under the condition of the same number of nodes, the network lifetimes in QSDN-WISE and SDN-DMRP are greater than that in SDN-WISE. SDN-DMRP adopts multi-path routing, which differentiates the energy consumption, and its network lifetime is therefore greater than that in SDN-WISE. The network lifetime in QSDN-WISE is significantly greater than that in SDN-DMRP and SDN-WISE because the clustering algorithm in QSDN-WISE considers the node residual energy when choosing cluster-head nodes, and adopts the idea of non-uniform clustering to balance the network energy consumption. Additionally, QSDN-WISE considers node residual energy and adopts multi-path routing based on double cluster heads, which further balances the energy consumption of the network.

VI. CONCLUSION

A hierarchical software-defined wireless sensor network architecture and a QoS-based clustering routing protocol called QSDN-WISE are proposed in this paper to solve the problems of poor adaptability and difficulty in network configuration and management, and support QoS in traditional network architecture for wireless sensor networks. The DCHUC clustering algorithm in QSDN-WISE adopts a non-uniform clustering mechanism to avoid an energy hole in the "hot spot" network area. The double cluster head mechanism with a low time delay and high reliability reduces the workload of a single cluster head, and implements QoS requirements for the data in clusters. Based on data classification, the centralized QSDN-WISE routing algorithm considers residual energy, node congestion degree, link stability, and distance between nodes as parameters to choose the next hop for nodes, and builds two heterogeneous forwarding paths for nodes, to meet the requirements of different data classes. Simulation results show that the QSDN-WISE clustering routing protocol can not only balance the energy consumption of WSN, but also provides QoS support for data with different QoS requirements, and compared with SDN-WISE, SDN-DMRP, and IRPL, QSDN-WISE exhibits good performance in energy saving, end-to-end delay, packet loss rate, and control of the number of messages.

REFERENCES

[1] G. Han, H. Wang, M. Guizani, S. Chan, and W. Zhang, "KCLP: A k-means cluster-based location privacy protection scheme in WSNs for IoT," *IEEE Wireless Commun. Mag.*, vol. 25, no. 6, pp. 84–90, Dec. 2018. doi: 10.1109/MWC.2017.1800061.

[2] W. Zhang, J. Wang, G. Han, X. Zhang, and Y. Feng, "A cluster sleep-wake scheduling algorithm based on 3D topology control in underwater sensor networks," *Sensors (Basel)*, vol. 19, no. 1, Jan. 2019, Art. no. E156.

[3] G. Han, H. Wang, J. Jiang, W. Zhang, and S. Chan, "CASLP: A confused arc-based source location privacy protection scheme in WSNs for IoT," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 42–47, Sep. 2018.

[4] G. Han, X. Miao, H. Wang, M. Guizani, and W. Zhang, "CPSLP: A cloud-based scheme for protecting source location privacy in wireless sensor networks using multi-sinks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2739–2750, Mar. 2019.

[5] I. Al-Anbagi, M. Erol-Kantarci, and H. T. Mouftah, "Priority- and delay-aware medium access for wireless sensor networks in the smart grid," *IEEE Syst. J.*, vol. 8, no. 2, pp. 608–618, Jun. 2014.

[6] W. Zhang, L. Liu, G. Han, Y. Feng, and Y. Zhao, "An energy efficient and QoS aware routing algorithm based on data classification for industrial wireless sensor networks," *IEEE Access*, vol. 6, pp. 46495–46504, 2018.

[7] H. Teng *et al.*, "A novel code data dissemination scheme for Internet of Things through mobile vehicle of smart cities," *Future Gener. Comput. Syst.*, vol. 94, pp. 351–367, May 2019.

[8] N. T. Long, M. P. Uwase, J. Tiberghien, and K. Steenhaut, "QoS-aware cross-layer mechanism for multiple instances RPL," in *Proc. Int. Conf. Adv. Technol. Commun.*, Oct. 2013, pp. 44–49.

[9] O. Gaddour, A. Koubâa, and M. Abid, "Quality-of-service aware routing for static and mobile IPv6-based low-power and lossy sensor networks using RPL," *Ad Hoc Netw.*, vol. 33, pp. 233–256, Oct. 2015.

[10] C. H. Barriquello, G. W. Denardin, and A. Campos, "A geographic routing approach for IPv6 in large-scale low-power and lossy networks," *Comput. Elect. Eng.*, vol. 45, pp. 182–191, Jul. 2015.

[11] G. Han, H. Guan, J. Wu, S. Chan, L. Shu, and W. Zhang, "An uneven cluster-based mobile charging algorithm for wireless rechargeable sensor networks," *IEEE Syst. J.*, to be published. doi: 10.1109/JSYST.2018.2879084.

[12] J. Li *et al.*, "Battery-friendly based relay selection scheme to prolong lifetime for sensor nodes in Internet of Things," *IEEE Access*, vol. 7, no. 1, pp. 33180–33201, 2019.

[13] W. Zhang, G. Han, J. Wang, and Y. Liu, "A BP neural network prediction model based on dynamic cuckoo search optimization algorithm for industrial equipment fault prediction," *IEEE Access*, vol. 7, pp. 11736–11746, 2019.

[14] X. Liu, T. Qiu, and T. Wang, "Load-balanced data dissemination for wireless sensor networks: A nature-inspired approach," *IEEE Internet Things J.*, to be published. doi: 10.1109/JIOT.2019.2900763.

[15] W. Zhang, L. H. G. Li, and L. Zhang, "E2HRC: An energy-efficient heterogeneous ring clustering routing protocol for wireless sensor networks," *IEEE Access*, vol. 5, pp. 1702–1713, 2017.

[16] C.-C. Lin, D.-J. Deng, Z.-Y. Chen, and K.-C. Chen, "Key design of driving industry 4.0: Joint energy-efficient deployment and scheduling in group-based industrial wireless sensor networks," *IEEE Commun. Mag.*, vol. 54, no. 10, pp. 46–52, Oct. 2016.

[17] J. Tan *et al.*, "A low redundancy data collection scheme to maximize lifetime using matrix completion technique," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, no. 5. doi: 10.1186/s13638-018-1313-0.

[18] X. Liu, "Node deployment based on extra path creation for wireless sensor networks on mountain roads," *IEEE Commun. Lett.*, vol. 21, no. 11, pp. 2376–2379, Nov. 2017.

[19] S. Sezer *et al.*, "Are we ready for SDN? Implementation challenges for software-defined networks," *IEEE Commun. Mag.*, vol. 51, no. 7, pp. 36–43, Jul. 2013.

[20] I. F. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou, "A roadmap for traffic engineering in SDN-OpenFlow networks," *Comput. Netw.*, vol. 71, pp. 1–30, Oct. 2014.

[21] Q.-Y. Zuo, M. Chen, G.-S. Zhao, C.-Y. Xing, G.-M. Zhang, and P.-C. Jiang, "Research on OpenFlow-based SDN technologies," *J. Softw.*, vol. 24, no. 5, pp. 1078–1097, May 2013.

[22] W. Ejaz, M. Naeem, M. Basharat, S. Kandeepan, and A. Anpalagan, "Efficient wireless power transfer in software-defined wireless sensor networks," *IEEE Sensors J.*, vol. 16, no. 20, pp. 7409–7420, Oct. 2016.

[23] R. Huang, X. Chu, J. Zhang, and Y.-H. Hu, "Energy-efficient monitoring in software defined wireless sensor networks using reinforcement learning: A prototype," *Int. J. Distrib. Sensor Netw.*, vol. 11, no. 10, Oct. 2015, Art. no. 360428. doi: 10.1155/2015/360428.

[24] D.-J. Deng *et al.*, "IEEE 802.11ax: Highly efficient WLANs for intelligent information infrastructure," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 52–59, Dec. 2017.

[25] T. Luo, H.-P. Tan, and T. Q. S. Quek, "Sensor OpenFlow: Enabling software-defined wireless sensor networks," *IEEE Commun. Lett.*, vol. 16, no. 11, pp. 1896–1899, Nov. 2012.

[26] A. Mahmud and R. Rahmani, "Exploitation of OpenFlow in wireless sensor networks," in *Proc. Int. Conf. Comput. Sci. Netw. Technol.*, Dec. 2012, pp. 594–600.

[27] T. Miyazaki *et al.*, "A software defined wireless sensor network," in *Proc. Int. Conf. Comput. Netw. Commun.*, Feb. 2014, pp. 847–852.

[28] L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "SDN-WISE: Design, prototyping and experimentation of a stateful SDN solution for wireless sensor networks," in *Proc. IEEE Conf. Comput. Commun.*, Apr./May 2015, pp. 513–521.

[29] M. Feng, S. Mao, and T. Jiang, "Enhancing the performance of futurewireless networks with software-defined networking," *Frontiers Inf. Technol. Electron. Eng.*, vol. 17, no. 7, pp. 606–619, Jul. 2016.

[30] D. D. Paolo *et al.*, "Exploiting state information to support QoS in software-defined WSNs," in *Proc. Ad Hoc Netw. Workshop*, Jun. 2016, pp. 1–7.

[31] W. Junfeng, M. Yiming, Z. Ping, M. S. Hossain, and S. M. M. Rahman, "A software defined network routing in wireless multihop network," *J. Netw. Comput. Appl.*, vol. 85, pp. 76-83, May 2017.

[32] G. Han, X. Yang, L. Liu, S. Chan, and W. Zhang, "A coverage-aware hierarchical charging algorithm in wireless rechargeable sensor networks," *IEEE Netw.*, to be published. doi: 10.1109/MNET.2018.1800197.

[33] X. Liu and P. Zhang, "Data drainage: A novel load balancing strategy for wireless sensor networks," *IEEE Commun. Lett.*, vol. 22, no. 1, pp. 125–128, Jan. 2018.

[34] M. Huang, W. Liu, T. Wang, H. Song, X. Li, and A. Liu, "A queuing delay utilization scheme for on-path service aggregation in services oriented computing networks," *IEEE Access*, vol. 7, pp. 23816–23833, 2019.

[35] L. M. Feeney, "An energy consumption model for performance analysis of routing protocols for mobile Ad Hoc networks," *Mobile Netw. Appl.*, vol. 6, no. 3, pp. 239–249, Jun. 2001.

[36] W. Zhang, G. Han, Y. Feng, and J. Lloret, "IRPL: An energy efficient routing protocol for wireless sensor networks," *J. Syst. Archit.*, vol. 75, pp. 35–49, Apr. 2017.

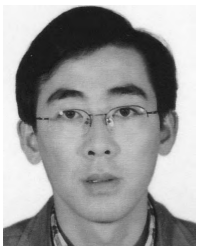
[37] P. Karkazis, P. Trakadas, H. C. Leligou, L. Sarakis, I. Papaefstathiou, and T. Zahariadis, "Evaluating routing metric composition approaches for QoS differentiation in low power and lossy networks," *Wireless Netw.*, vol. 19, no. 6, pp. 1269–1284, Aug. 2013.

[38] *SDN-WISE-API*. Accessed: Jun. 10, 2018. [Online]. Available: https://www.mvnjar.com/com.github.sdnwiselab/sdn-wise-api/1.0.7/detail.html

XIAOBO TAN received the B.E. degree from Liaoning Normal University, in 2000, and the M.E. degree from Northeast University, China, in 2006, where he is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering. He is currently an Associate Professor with the School of Information Science and Engineering, Communication and Network Institute, Shenyang Ligong University. His research interests mainly include wireless sensor networks and embedded systems.

HAI ZHAO received the Ph.D. degree in computer application technology from Northeastern University, China, in 1985. He is currently a Professor and a Doctoral Supervisor with the School of Computer Science and Engineering, Northeastern University. He established the International Association for Internet Data Analysis (CAIDA), China's First Node, and maintained long-term research and teaching cooperation with the University of Waterloo, Canada, UTD, USA, and UTA universities. He has applied for ten national patents and software copyrights, and published over 200 papers in related international conferences and journals. His current research interests include embedded systems, sensor networks, and pervasive computing.

GUANGJIE HAN (S'03–M'05–SM'18) received the Ph.D. degree from Northeastern University, Shenyang, China, in 2004. From 2004 to 2005, he was a Product Manager with ZTE Company. From 2005 to 2006, he was the Key Account Manager with Huawei Company. In 2008, he finished his work as a Postdoctoral Researcher with the Department of Computer Science, Chonnam National University, Gwangju, South Korea. From 2010 to 2011, he was a Visiting Research Scholar with Osaka University, Suita, Japan. In 2017, he was a Visiting Professor with City University of Hong Kong, Hong Kong. He is currently a Professor with the Department of Information and Communication System, Hohai University, Changzhou, China, and a Distinguished Professor with the Qingdao University of Science and Technology, Qingdao, China. He is the author of over 330 papers published in related international conference proceedings and journals, including the IEEE COMST, IEEE TMC, IEEE TIE, IEEE TII, IEEE TCC, IEEE TPDS, IEEE TVT, IEEE TETC, IEEE IoT Journal, IEEE TETCI, IEEE Systems Journal, IEEE Sensors Journal, IEEE Wireless Communications, *IEEE Communications Magazine*, and IEEE Network and the holder of 125 patents. His H-index is 32 and i10-index is 90 in Google Citation (Google Scholar). Total citation of his papers by other people is more than 4426 times. His current research interests include the Internet of Things, the industrial Internet, machine learning and artificial intelligence, mobile computing, security, and privacy. He received the ComManTel 2014, ComComAP 2014, Chinacom 2014, and Qshine 2016 Best Paper Awards. He has served as the Co-chair for more than 50 international conferences/workshops and as a Technical Program Committee Member of more than 150 conferences. He has served on the editorial boards of up to 16 international journals, including the IEEE JSAC, IEEE Network, IEEE Systems Journal, IEEE Access, IEEE/CCA JAS, and *Telecommunication Systems*. He has guest edited a number of special issues in IEEE journals and magazines, including the *IEEE Communications Magazine*, IEEE Wireless Communications, IEEE Transactions on Industrial Informatics, and *Computer Networks*. He has served as a Reviewer for more than 60 journals.



WENBO ZHANG received the Ph.D. degree in computer science and technology from Northeastern University, China, in 2006. He is currently a Professor with the School of Information Science and Engineering, Shenyang Ligong University, China. He has published over 100 papers in related international conferences and journals. His current research interests are ad hoc networks, sensor networks, satellite networks, and embedded systems. He received the ICINIS 2011 Best Paper Awards and up to nine Science and Technology Awards, including the National Science and Technology Progress Award, and the Youth Science and Technology Awards from the China Ordnance Society. He has served on the editorial boards of up to ten journals, including the *Chinese Journal of Electronics* and the *Journal of Astronautics*.

TENG ZHU received the B.S. degree in computer science and technology from Shenyang Ligong University, China, in 2016, where he is currently pursuing the master's degree with the School of Information Science and Engineering. His current research interest includes software-defined network (SDN)-based network management technology for wireless sensor networks.