IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 19, 2019, accepted November 25, 2019, date of publication December 2, 2019,
date of current version December 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2956995

Self-Adjustable Active Sequence to Reduce Latency in Duty Cycle WSNs

XUAN LONG[1], WEI LIU[2], TIAN WANG[3], ZHIWEN ZENG[1], AND MING MA[4]
[1]School of Computer Science and Engineering, Central South University, Changsha 410083, China
[2]School of Informatics, Hunan University of Chinese Medicine, Changsha 410208, China
[3]College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China
[4]Department of Computer Science, Stony Brook University, Stony Brook, NY 11794, USA

Corresponding author: Wei Liu (weiliu@csu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61772554, Grant 61572528, and Grant 6157256.

ABSTRACT The wireless sensor networks (WSNs) have a great application prospect, which is composed of much simple hardware and small-size sensor nodes to realize the perception of the surrounding environment. It is often widely used that nodes work in a duty cycle mechanism of periodic sleep and awakening in WSNs. However, this mechanism also increases latency and the routing table length whereas saving energy. In this paper, the Self-adjustable Active Sequence (SAC) scheme is proposed to solve the above problems. It enables optimization of energy, latency and routing tables. The main innovations are as follows. Firstly, the SAC scheme divides the continuous active slots into multiple shorter slots to reduce the latency. Second, SAC scheme adds more active slots by using the remaining energy. Third, decreasing the number of the forwarding nodes to reduce the routing table length. In this case, the optimization of delay only brings small gains, so we should focus on the improvement of routing. It has a better performance in reducing the delay and reducing the length of the routing table through both theoretical analysis and experimental results. If and only if the continuous active slots are divided into several segments, the delay is reduced by 33%. And the number of active slots by using the far sink region energy further reduce the delay by 29% in general. What's more, when reducing the size of the forwarding nodes set, the routing table length is further reduced by 29%.

INDEX TERMS Duty cycle, latency, lifetime, routing table length, wireless sensor network.

I. INTRODUCTION

With the development of microprocessor technology, more and more intelligent sensing devices have been applied to various applications, which greatly expanding their application field [1]–[4]. Wireless sensor network (WSNs) are essential elements for realizing the Internet of Things (IoTs) [5]–[8], which can be widely used in industrial production sites [9], [10] traffic information [11], [12], crop monitoring [13]–[15], medical monitoring [16], [17], personal health monitoring of personal wearable devices and other applications [18], [19]. Wireless sensor nodes have been extensively developed due to their small size, low cost, and flexible deployment. Besides, as a result of the development of electronic technology in recent years, the computing, storage and communication capabilities of the sensing devices have been significantly improved. It is not inferior to the processing power of personal computers more than a decade ago. These sensing devices are numerous, making universal sensing possible [20]–[24].

There are two critical issues in wireless sensor networks: (1) one is low-latency data routing [25]–[27], (2) the other one is energy efficiency [7], [9], [14], [19], [25], [28]–[32]. The most crucial task of wireless sensor networks is to perceive the surrounding environment and get the state data of the monitored object [4], [5], [9], [11], [12], [14], [20], [28], [33]–[36]. Then, sensing data is routed to the sink or control center via multi-hop routing. The control center reacts and processes the monitored objects after analyzing and extracting the sensing data [33]–[36]. Therefore, in order to process the detected events promptly, it is an extremely significant research issue to route sensing data to sink quickly [25]–[27]. On the other hand, the sensor network is battery-powered [7], [9], [14]. Considering the economic and the convenience of use, the structure of the sensor node

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

is relatively simple, and its volume is small, which determines that both its battery and capacitance are low. As a result, the energy of sensor nodes is minimal [25], [28]. How to use energy effectively and extend the lifetime of the network is another vital research issue [9], [26], [36]. Sensor nodes adopt the duty cycle working mode to save energy, which is a practical way to save energy [25], [26], [37], [38]. In the duty cycle mode, the nodes awake and sleep periodically [25], [26], [37], [38]. Since the energy consumption of nodes in the sleep state is 1/100 or even 1/1000 of that in the active state, it is superior for nodes to be sleepy to save energy as far as possible. However, when the node is sleepy, it cannot sense, receive and send data [25], [26], [37], [38]. Therefore, when the sender node has data to send, but the receiver node is in sleep state, the sender node must wait for the receiver node to wake up before data routing. Thus, the duty cycle mechanism will cause higher latency in data routing and consume a lot of energy by cause of the sender node idle waiting [25], [26], [37].

In order to avoid wasting energy in vain when the sender needs to send packet, then an effective routing scheduling strategy has been proposed, which is to find its own route according to the active slots of the forwarding nodes. That is to say, each node stores a routing table, which stores the minimum delay to reach the next hop and the relay node selected at each slot [38]. In this way, each node can know its relay node and slot to send packets by looking up the routing table when it has data transmission [38]. Just like that, if it needs to wait for more than one slot to send data, it is not necessary to wait for its relay node to wake up. Instead, it goes to sleep, waits for a slot that can forward data to arrive and then wakes up to transmit data, which can save energy [38].

In such a forwarding method by routing table, there are also two critical issues to be studied. One is how to reduce latency, and the other is how to reduce the routing table length. As mentioned above, in duty cycle based WSNs, when the sender node has data to send, its forwarding nodes may all be in the sleep state. In this case, the sender node needs to wait for one of the forwarding nodes to wake up before sending data, so the time of the packet stays in sender node is called the latency, or the single-hop latency [25], [26], [37], [38]. In addition, the sum of all latency, which has passed through multiple hops from the source node to the sink node is called the end to end latency. The routing table stores the next hop node selected by the node, the slot in which the data is sent, as well as the latency to the next hop [38]. The longer the routing table is, the more storage space is required, and the more time it takes to look up the routing table.

However, the issue for designing a routing strategy to implement small latency and shorter routing table length whereas making energy efficient in duty cycled WSNs [38]. Although some studies have attempted to reduce latency and routing table length, there is still room to improve routing performance further. Therefore, in this paper, the self-adjustable Active Sequence (SAC) scheme is proposed to reduce latency in duty cycled wireless sensor networks. Compared with the previous work, the main innovation points of this paper are as follows:

(1) Firstly, SAC scheme divides only one-segment continuous active slots into multiple continuous active slots with smaller length, which can effectively reduce the latency. In the previous strategy, since the active slots is continuous, if the sender node happens to have data transmission at the range of the continuous active slots, there is no need to wait. On the contrary, if it is not within the range of the continuous active slots, it needs to wait for the node's active slot to arrive before sending, which generates a high latency. After dividing the continuous active slots into multiple active slots, the latency can be effectively reduced without increasing energy consumption of nodes.
(2) Secondly, SAC scheme makes full use of the remaining energy of nodes in the far sink region to increase active slots. And through these added active slots, some segments of active slots are connected into continuous active slots, thereby reducing the number of active slots segments, thus achieving the purpose of reducing both latency and routing table length.
(3) Thirdly, it is found that when there are enough active slots or forwarding nodes, the latency is already minimal. At this moment, if some forwarding nodes do not participate in the routing, there will be no increase in latency. Instead, the routing table length can be commendably reduced because of the reduced relay nodes. Therefore, this paper proposes a method to reduce the routing table length by reducing the size of forwarding nodes.
(4) The SAC scheme proposed in this paper has better performance than previous strategies through sufficient theoretical analysis and simulation results. Under the condition of Experiment two in this paper, when only the continuous active slots are segmented, the latency can be reduced by 33%. And when using the far sink region energy to add the active slots, the delay can be further reduced by 29%. What's more, when the size of the forwarding node set is reduced, the routing table length can be further reduced by 29%.

The rest of this paper is organized as follows: Section ''Related work'' mainly introduces the research associated with this paper. Section ''The system model and problem statements'' mainly shows the related model used in this paper. Section ''The design of Self-Adjustable Active Sequence (SAC) scheme'' gives the SAC scheme and related specifications. In section ''Parameter optimization and performance analysis'' we investigate the performance analysis of SAC scheme from the angle of delay and routing. Section ''analysis of experimental results'' presents the theoretical analysis and simulation results of two experiments. Section ''Conclusion'' provides a summary of the paper and the future research direction.

TABLE 1. The related papers, achievements and limitations.

number	achievements	limitations
[4] [7]	low delay	non-dynamically generated data
[1][13][26][32]	widely used	compete channels
[41]	reliability guarantee	large delay
[9]	balance the energy	long routing table
[42]	no congestion	
[43]-[45][46]	multipath routing	high energy consumption
[48]	data aggregation on the ring	
[47]	data aggregation, low delay	
[7][33]	cluster head	
[25][26][37][38]	duty cycle	large delay
[49]	low delay	more hops
[50]	self-adaptive	
[38]	small routing length	large delay
[51]	continuous slots	large delay

II. RELATED WORK

Fast data collection in wireless sensor networks has been the essential issue in its research [4], [5], [9], [11], [12], [14], [20], [28], [33]–[36], [39], [40]. In fact, the time required for data collection varies owing to different data collection strategies. According to the different MAC protocol used in data collection, it can be divided into contention-based data collection mechanisms [1], [9], [13], [26], [32], [36] and non-contention based data collection mechanisms [4], [7]. The non-contention MAC protocol is represented by the Time Division Multiple Access (TDMA) protocol [4], [7]. Most of the data collection protocols adopt are contention-based protocols [1], [9], [13], [26], [32], [36]. The main feature of the non-contention data collection protocol represented by the TDMA protocol is that the time is divided into units called slots. A slot time can send or receive a packet [4], [7]. Moreover, in most studies, a time slot also includes the time required to establish a communication link. In the non-contention MAC protocol, slots are only scheduled when it needs to transmit data, that is, the node is active only when the node receives and sends the data. And when there is no data transmitted, the node goes to sleep to save energy. Since the energy consumption in the sleep state is several orders of magnitude lower than that in the active state, the TDMA protocol has excellent performance in terms of energy consumption [4], [7]. Meanwhile, since the scheduling algorithm for data collection can be optimized in advance, each node can schedule the active slots accurately to minimize the time required for network data collection, and thus the delay is relatively small. However, the most significant limitation of non-contention strategies such as TDMA is that the time of the whole network needs to be synchronized, so it is essential to pay the cost of time synchronization [4], [7]. What is more, the node's data operation must be strictly carried out in accordance with the slot planned. Therefore, when the node generates more packets than expected, the newly added packets cannot be transferred to sink because there are no active slots available [4], [7]. All in all, this method is not suitable for networks that generate data dynamically [4], [7].

Conversely, in the contention that based collection mechanism, the node needs to contend for a channel once it has data to send. After the channel competition is successful, data transmission is performed. As a result, there are fewer network requirements in this protocol, which can be applied to various networks. However, the main disadvantage of such method is that it requires competing channels between multiple nodes, whereby the performance in terms of energy consumption and delay is not as good as the TDMA protocol [1], [9], [13], [26], [32], [36]. Some research work related to this paper is discussed in detail below.

The related papers discussed in this paper are as Table 1, so that readers feel no trouble during reading.

A. RESEARCH ON DELAY OPTIMIZATION

Delay is an important and widely studied performance indicator in wireless sensor networks [25]–[27]. There is no doubt that delay has different performance in different data routing strategies.

(1) Optimization of data routing delay in the contention MAC mechanism. It is well known that many routing strategies have been proposed based on contention MAC mechanisms [1], [9], [13], [26], [32], [36]. The main factors considered in the routing strategy include energy consumption, delay, reliability, and so on. The earliest routing strategy proposed is the shortest routing algorithm. In the shortest routing algorithm, when a node has a packet to send, it searches for the node nearest to the sink within its own transmission radius and forwards the packet as relay node. This method always selects the node nearest to sink for routing, so the routing formed is the shortest way to sink. The shortest routing algorithm generally assumes that the node is always active, so the delay of this method is relatively small.

However, this method only considers the distance from the sink without considering the quality of the communication link, energy consumption, congestion, and so on. Therefore, in the case of the shortest route, the delay and lifetime are not necessarily can achieve the minimum. In the network with poor link communication quality, the probability of successful transmission of packets is not 100%, but a probability of less than 100%, only 80%-90%. Therefore, it is necessary to adopt some reliability routing mechanisms to guarantee data transmission, which will also have a great impact on delay [41]. The simplest method to ensure the reliability of data transmission is the Send and Wait Automatic Repeat-Qequest (SW-ARQ) protocol. In SW-ARQ protocol, after the sender sends the packet, it begins to wait for the receiver to return the ACK message to confirm that the packet has been received [41]. If the sender receives the ACK message at the specified time, it continues to send the next packet.

Otherwise, the packet is resent until the wait time reaches the timeout threshold. The above process continues until the packet is sent successfully, or the packet is aborted due to the number of retransmissions reaches the preset threshold. In such a process, the delay of data transmission is higher than that of single data transmission [41].

Although the shortest routing algorithm forms the shortest route, the design of the routing algorithm also needs to consider the node's energy consumption and network lifetime in practice. In many networks, adopting the shortest routing algorithm will cause excessive energy consumption of some bottleneck nodes and the surplus energy of some nodes [9]. Therefore, an energy consumption balanced routing strategy is proposed. In such a routing method based on energy consumption balance, the criterion for the node to select relay node is not only the distance to the sink, but also for the residual energy, so as the two factors of energy consumption and distance are considered together [9]. Such a routing strategy can obtain a more extended network lifetime by balancing the energy consumption between nodes [9], even if it may have a longer routing length than the shortest routing algorithm and increases the delay.

Based on the above ideas, some routing choices include the degree of network congestion into the next hop of routing, to select those nodes with high residual energy, close to the sink, and uncongested links [42]. Obviously, choosing a path without congestion can reduce the delay of the route [42].

Besides, there are also some studies on the strategy of multipath routing for security [43]–[46]. The starting point of this routing strategy is that, when a single route is adopted, only if there is a malicious node on the routing path, the data packet will be cut off, which will cause harm to the network. Therefore, if several different routing paths are issued at the same time, the data transmission is successful if anyone of them is successful. In such a route, a packet will have many routes at the same time, result in its energy consumption is approximately many times of the original in theory, but the performance of delay is better [46].

(2) Delay Optimization in data aggregation networks. In the previous discussion, the node's packets do not change during the routing process. However, in many applications, data aggregation can be carried out when multiple data packets meet due to the correlation [4], [5], [7], [27], [34], [47], which integrates data packets into smaller ones than the original two. This method achieves the goal of reducing the amount of data transmitted and saving energy. As a result, a variety of data aggregation collection strategies are proposed. In these approaches, the emphasis is on how to make the data packets meet as much as possible so that make the data packets merge and reduce the network data volume the most [4], [5], [7], [27], [34], [47]. In the data aggregation routing strategy that mainly considers reducing the data volume, packets tend to change the routing direction of the shortest route and route to the area as much as possible to make the data aggregation, such as ring [48]. After data aggregation on the ring, it will be routed to sink. Data aggregation will reach the maximum value, whereas the delay is often an irrelevant consideration indicator [48].

In some studies, data aggregation and delay need to be considered simultaneously [47]. In order to aggregate the data as much as possible, if the node waits longer after receiving data packets, it can wait for more data packets to arrive, thus making more data packets aggregate into smaller packets. Therefore, from the perspective of data aggregation, it is desirable to make packets stay on the node for as long as possible, which enables data aggregation to play a larger role. However, if each packet stays in the node for a long time, the delay of data routing can be enormous. Li et al. proposed a comprehensive optimization method to optimize data aggregation and reduce delay [47]. The main points of the method are as follows: each node sets two thresholds: the aggregation deadline (T_t) and the aggregation threshold (N_t) [47]. T_t represents the maximum time that the packet stays in the queue, whereas N_t represents the maximum queue length of the node. Data is sent when the waiting time of the node in the queue reaches T_t or the queue length reaches N_t [47]. When selecting parent nodes for packet transmission, the node with the longest packet queue in the parent nodes should be selected as the relay node. After selecting such parent nodes, the packet will stay on it for a short time, thus reducing the delay of data routing [47].

The clustering network is also an effective method of data fusion [7], [33]. In a clustering network, the network is divided into a basic unit cluster [7], [33]. There is a special node called cluster head in a cluster, and other nodes are called cluster member nodes. Cluster member nodes send their own data to the cluster head. Then the cluster head makes data aggregation and sends it directly to the sink or sends it to sink through multi-hop routing between cluster heads [7], [33].

(3) Delay optimization of TDMA based data aggregation under the non-contention mechanism. The advantage of the TDMA scheduling strategy is that each node schedules slot for data operations in advance [7]. When the network runs, the node only needs to perform the corresponding data operations by the slots arranged in the scheduling algorithm. In this way, energy consumption operations such as channel conflict, channel detection, and idle waiting can be saved, and its data collection delay is also relatively small.

The data aggregation discussed above is a method of data aggregation rate between [0, 1]. Furthermore, there is a unique data aggregation method that can aggregate n data packets into one when they meet, which dramatically reduces the data number of nodes and save energy. It is for this reason that makes TDMA scheduling possible. Li *et al.* [7] proposed a novel data collection strategy for unequal clustering networks. The unequal clustering proposed by them is different from the unequal clustering in the previous strategies. In the previous unequal clustering, the clusters in the far sink region are large, whereas the clusters in the near sink region are small. On the contrary, the unequal clustering structure proposed by them is that the clusters in the near sink region

are large, whereas that in the far sink region is small. The reason why they proposed such a structure is that: in the first stage of cluster data collection, the operation of sending the cluster data to cluster head is completed simultaneously in the previous equal cluster radius data collection. The reason indicates that the data aggregation within the cluster is performed in parallel. In the second stage, inter-cluster data routing needs to be routed from the farthest cluster head to the sink layer by layer. Such operation is performed serially and therefore takes longer. However, after changing to the unequal cluster structure with a small radius in the far sink region and a large radius in the near sink region, the intra-cluster data aggregation in the far sink region is completed first, thereby initiating data routing to the sink. The clusters in the near sink region are large, and the data aggregation in the clusters takes a long time. When the distant data is routed to the clusters near the sink, the data in the clusters is just completed. In this way, whether it is intra-cluster data aggregation or inter-cluster data routing is performed in parallel, which can effectively reduce the time required for data collection.

(4) Duty cycle-based routing. The closest approach to this paper is the routing strategy in the duty cycle based WSNs [25], [26], [37], [38]. The duty cycle working mode of a wireless sensor network is adopted from the perspective of saving energy [25], [26], [37], [38]. Most of the strategies discussed above are based on networks where nodes are always active. However, in most sensor networks applications, the time unit for monitoring events and physical phenomena is sometimes measured in days, hours, and minutes, so there is no need to keep the nodes active perpetually. The energy consumption of the node in the active state is more than a hundred times that of the sleep state. Therefore, the node should be in the sleep state as much as possible to save energy. That is how the duty cycle works. In such a working mode, the time is divided into cycles. The cycle is divided into two states: active and sleep. The periodic active/sleep of the node can greatly save energy in the network without affecting the monitoring [25], [26], [37], [38].

However, this duty cycle mode increases the delay of data routing. In the previous non-duty cycle WSNs, the node is always in the active state, the delay is mainly caused by the queued delay of the packets when establishing communication connections and data transmission. Nevertheless, in the duty cycle based WSNs, a delay called latency is added. The main reason for the occurrence of latency is that the nodes adopt the working mode of the duty cycle. When the sender has data transmission, if all forwarding nodes are in sleep state, it has to wait for one of them to wake up and generates the wait delay [25]. The latency is related to the duty cycle adopted by the network. The general rule is: the longer the node is active, the smaller the latency is and vice versa. What is more, latency is also related to the node density of the network. The higher the node density is, the greater the number of forwarding nodes of the sender will be. Therefore, when the sender needs to send data, the probability of more than one forwarding node being active is high, and the latency is small [25].

There are also some studies proposed to reduce latency for duty cycle based WSNs. In some of them, one cycle is divided into n slots, only one slot is in the active state, whereas the other n-1 slots are in the sleep state. At this moment, the duty cycle of the node is $\frac{1}{n}$. The FFSC strategy proposed by Liu *et al.* [49] can effectively reduce latency. In the duty cycle based WSNs, when the sender wants to send data, it also chooses the node closest to the sink to make the distance from one hop to sink furthest. Thus, the hops needed to route to the sink are minimized, which saves energy and reduces delay [49]. However, in this case, the node closest to the sink among its forwarding nodes is not necessarily in the active state. The sender has two choices, and one is to wait for the node closest to the sink to wake up, the other is to choose the node closest to the sink from among forwarding nodes in the active state as the relay node for data routing. The waiting time in the first selection is uncertain, and it also consumes energy while increasing latency. In the second choice, the node selected may be very far away from the sender, so the distance of the 1-hop route to sink is small. Therefore, more hops are required to route to sink, which leads to more massive delay and higher energy consumption. In the FFSC strategy, there is a circumstance that the nodes in the near sink region consume more energy, whereas these in the far sink region have surplus energy in WSNs. In the case, in the near sink region, the sender sends data once there are active forwarding nodes, and in the far sink region, the sender waits for the node closest to the sink among the forwarding nodes to wake up and sends data, which can effectively reduce the delay.

In the above research, the duty cycle of the node is random and does not require synchronization. That eliminates the need for the system to maintain synchronization, so its cost is relatively small. However, this asynchronous method may cause the node closest to the sink among forwarding nodes to be in the sleep state when the sender has data transmission. This problem will not exist. This problem does not exist if the synchronization mechanism is adopted.

Other researchers proposed an adaptive duty cycle to reduce delay. In some studies, the size of the duty cycle is determined according to the amount of data undertaken by the node. When the node undertakes a large amount of data, a larger duty cycle is adopted, whereas when the node undertakes a small amount of data, a lower duty cycle is adopted to save energy. Chen *et al.* [50] believe that this method looks well enough, but it may make the network performance worse in applications. This is because the nodes in the far sink region assume a small amount of data, whereas the nodes in the near sink region assume a large amount of data in WSNs. Previous research results show that the duty cycle of nodes in far sink region should be small, whereas that of nodes in near sink region should be large. This setting makes the nodes in the far sink region consume less energy and have a surplus. Despite the fact, the small duty cycle

can save energy consumption while increasing the latency. However, since the energy of these remote sink nodes is redundant, there is no sense of saving energy, which increases the latency by reducing the duty cycle. Therefore, the self-adaptive duty-cycled strategy proposed by Chen *et al.* [50]. In their proposed strategy, the duty cycle of the node in the far sink region is larger than that of the node in the near sink region. This allows the network to take full advantage of the residual energy of these nodes while reducing latency without reducing network lifetime. But the nodes near the sink region adopt an appropriate duty cycle can save energy and improve network lifetime. In this way, the purpose of reducing the latency and maintaining the high network lifetime is achieved [26], [50].

In the previous studies, the duty cycle of nodes is usually divided into two segments, one is the active time slots of nodes, and the other is the sleep time slots. That is to say, the active time and sleep time are continuous within one cycle [25]. Shahzad *et al.* [38] used routing tables to data routing. It is recorded that the next-hop node which can minimize the latency and the latency which reaches the sink when a node has data transmission in any slot. The critical point is that each node updates its routing table by receiving the slot information and routing table information from the neighbor nodes, so that the node can clearly know the slots and latency of routes. The main target of their research is how to reduce the routing table length effectively. Because reducing the length of the routing table can reduce the time required by the node to find the route and help reduce the delay [38].

In [51], a method is proposed to spread the continuous active slots in the duty cycle to the entire cycle to improve the monitoring performance. It is noticed that if the node's active time slots are consecutive, then the subsequent long time is a period of sleep slots. Consequently, the nodes fail to monitor the environment for a long time, which will cause a large monitoring delay. In brief, dispersing continuous active slots into the whole cycle can productively reduce the monitoring delay [51].

B. RESEARCH ON LIFETIME

Improving network lifetime is one of the most critical factors in the design of routing strategies [9], [26], [36]. By the node energy is highly limited, how to save the energy to improve the network lifetime is a significant issue in WSNs. Up to now, there have been quite a lot of researches on saving node energy. However, there is an unusual phenomenon called ''energy hole'' in the wireless sensor networks, which makes the research on improving network lifetime have a challenge. ''Energy hole'' refers to a phenomenon that the premature death of a network due to the energy exhaustion of nodes within one hop of the sink. This is because all data in the wireless sensor networks must be routed to the sink, and the nodes within one hop of the sink handle all data in the whole network, so its energy consumption is the highest [9]. When the nodes within one hop of the sink run out of its energy and die, all data in the network cannot be sent to the sink, leading to network death [9]. According to relevant studies, due to the influence of the energy hole, there is still up to 80% energy remains in the network when the network dies [9].

For the duty cycle based WSNs, the more active slots of a node, the higher energy consumption and the smaller lifetime. Therefore, from the point of view of improving the lifetime of the network, there should be fewer active slots in a cycle. On the other hand, the smaller active slots, the higher the latency of data routing. Shahzad et al. proposed an optimized routing strategy based on the routing table [38]. Since the number of active slots is not changed, the reduction of latency is also limited. This paper proposes a method to add active slots through residual energy, which can commendably reduce the latency.

Wireless sensor networks are a vital component of the Internet of Things [52], [53] and the most primary source of data collection. The combination of data and artificial intelligence makes the current network to reach a new horizon [54], [55]. With the development of sensor devices, its quantity reaches a huge scale, which makes the current network develop towards marginalization. Edge networks and Fog computing are the manifestations of this development trend [53], [56]. Therefore, it is of considerable significance to study the data routing strategy with effective energy and low latency.

III. THE SYSTEM MODEL AND PROBLEM STATEMENT

A. THE NETWORK MODEL

The network model adopted in this paper is composed of many static, identical, and randomly deployed nodes. Node random deployment means that nodes are deployed based on the location of the data to be monitored and the communication link conditions in the actual situation, without specifying the location of each node. Because the specific location of the nodes is not involved in this problem, the node density has no significant effect on the experimental results, and no special discussion is needed. When the network is initialized, the deployed nodes form a multi-hop network through self-organization, which is dynamic and self-adjusting. Nodes consist of a sink node and some ordinary nodes. All ordinary nodes transmit data to sink node, and sink node receives data from all other nodes through multiple hops.

The sink node has unlimited power, whereas batteries drive ordinary nodes with limited power. The ordinary nodes adopt awake/sleep periodic rotation mode to save energy. Divide a period into multiple time slots and use N to represent the number of time slots in a period. Number all time slots starting from 1 in ascending order, so the slots are represented by $\{1, 2, \ldots, N-1, N\}$. Each node selects a part of continuous slots as active slots, and M represents the number of active slots. In the routing initialization phase, each node randomly chooses its own starting active slot, all sensor nodes are awake, and the routing table is empty. Sink node starts to broadcast the routing broadcast packets, and when other nodes receive the routing broadcast packets, processing and

forwarding them. The routing table is constructed on each node in a distributed manner. When a node is unavailable, then the routing table is updated, so the network topology is dynamic.

B. THE ENERGY CONSUMPTION MODEL

In this network model, there are three types of energy consumption. Firstly, the energy consumed when sending the packet. Secondly, the energy consumed when receiving the packet. Thirdly, the energy consumed by the sender waiting for sending data or the receiver waiting for data reception. Assume that the transmitted packets are the same size, so the energy used to send and receive packets remains the same. For the energy generated by waiting, this paper adopts a scheme to reduce the number of active slots, that is, the node adopts the mode of awake/sleep periodic rotation to reduce the energy consumed.

In addition, there is an imbalance in energy consumption in the wireless sensor network. The nodes near the sink are responsible for the forwarding of the remote nodes' data, so the energy consumption is much higher than that in the remote sink area. In this way, the energy hole phenomenon will lead to the death of nodes within one hop of the sink and the death of the whole network in advance. Therefore, a large amount of residual energy in the remote sink region can be fully utilized to reduce latency without reducing lifetime effectively.

C. PROBLEM STATEMENT

Node adopts the duty cycle mechanism, but it also brings high delay to data routing and increases the routing table length, which will bring node with greater storage space requirements and longer searching time for routing table. Therefore, the goal of this paper is to minimize the delay and ensure the minimum routing information on the premise of not damaging network lifetime.

The delay consists of two parts: the transmission delay and the wait delay. The wait delay can be expressed by the latency, and it indicates the time it takes for the packet in the sender be ready to be sent out. The transmission delay represents the time it takes for the data to be transmitted from the beginning to the receiver. It is assumed that the transmission delay to the next hop is one time slot, and it can complete the transmission or reception of a packet in one slot., so the delay is the latency plus 1. Thus, the main factor affecting delay is latency, lowering the delay is to reduce the latency. In this paper, except when calculating the value, delay is always 1 greater than latency. In other parts of the paper, the two mean the same meaning and can be replaced with each other.

Definition 1: Active slots coverage. The active slots coverage of single node j is T_j^c, and its distribution satisfies Eq. (6). So, the calculation formula of the active slots coverage of multiple forwarding nodes $\{1..j\}$ is shown as Eq. (1):

$$T^c_{\{1..j\}} = \bigcup T_j^c \,|\, j \text{ is in forwarding node} \{1..j\} \tag{1}$$

Definition 2: Non-increasing Delivery-latency Interval (NDI). A series of delay non-incremental slots is called an NDI. The NDI-based routing scheme is adopted in this paper. The delay in an NDI satisfies Eq. (2):

$$d_i^t = \begin{cases} d_i^{t-1} - 1, & d_i^{t-1} > 1 \\ 1, & d_i^{t-1} = 1 \end{cases} \tag{2}$$

where d_i^t represents the delay of node i at slot t, t represents the time slot, and i represents the node id. This above equation gives an algorithm for calculating the delay of the next slot based on the delay of the previous slot in a series of delay non-incremental slots.

Definition 3: Routing table storage format. Figure 1 shows the storage structure of the node routing table. Each route record is an NDI. The routing table consists of four items: starting slot, starting delay, ending delay, and next hop, where starting slot represents the first slot of this NDI, the starting delay represents the delay calculated by the starting slot, the ending delay is the last slot of a NDI and is always 1, and the last item next hop represent the node selected as the next hop relay node. Its advantage is that we only need to know a starting time slot and the delay of starting time slot to know the delay of any time slot in this NDI.

Starting slot	Starting delay	Ending slot	Next hop

FIGURE 1. Routing table: NDI-based routing scheme.

Definition 4: Routing table storage principle. The delay non-incremental. That is to say, the delay at slot $t+1$ is always one less than or equal to the delay at slot t in a route record. That is to satisfy $d_i^t \geq d_i^{t+1}$. Only when the delay at slot t is 1, the delay at slot $t+1$ is equal to that of slot t. It also satisfies the following property: the starting slot of the first routing record is always 1, and the ending delay of all routing records is 1 except the last routing record.

For the convenience of readers, notations used in the paper are listed in Table 2, where the column named "Notation" contains the names of variables, and their definitions are presented in the column "Definition".

IV. THE DESIGN OF SELF-ADJUSTABLE ACTIVE SEQUENCE (SAC) SCHEME

A. RESEARCH MOTIVATION

In this section, we through a concrete example to illustrate the motivation of the Self-adjustable Active Sequence (SAC) scheme. The network topology model adopted is shown in Figure 2. Figure 2 represents the topology diagram through self-organization. In order to facilitate the calculation, each node is numbered sequentially from the outside to the inside. In such a network, there are 25 nodes, and K is used to represent the number of nodes in the network, that is, $K = 25$. The nodes are represented as {A, B, C, . . ., W, X, Y}, S node is the sink node.

TABLE 2. Notations used in the paper.

Notation	Definition
v_i	The node identification of node i
M	The number of active slots in a period
N	The number of time slots in a period
T_j^c	The active slots coverage of single node j
$T_{\{1..j\}}^c$	The active slots coverage of nodes $\{1..j\}$
T_i	The starting slot set in RT^i
$T_{i,j}$	The starting slot set in $RT^{i,j}$
T_i^k	The k-th slot in T^i
d_i^t	The delay of node i at slot t
$d_{i,j}^t$	The delay of node i to j at slot t
$\overline{d_i}$	The average one-hop delay of node i
$\Delta\overline{d_i}$	The reduced average one-hop delay of node i
$s_{i,j}^t$	The slot number of the earliest active slot in forwarding node j after slot t
$\mathcal{D}_i$	The sum of the average one-hop delay on the route path from node i to the sink
$\Delta\mathcal{D}_i$	The reduced $\mathcal{D}_i$
RT_i	The routing table of node i
$RT_{i,j}$	The routing table from node j to i
RT_i^k	The k-th routing record in RT_i
RBP_j	The routing broadcast packet of node j
s_j	The starting active slot of node j
$s_j[]$	The starting active slot array of node j
s_i^d	The random partition slot in the active slots of node i
s^a	The number of starting active slots distribution
e_j	The ending active slot of node j
$e_j[]$	The ending active slot array of node j
e_j^k	The k-th slot in $e_j[\,]$
s_k^a	The k-th smallest slot in the starting active slots set
e_k^a	The ending active slot corresponding to s_k^a
$v[\,]$	The node identification array
v_i^{ne}	The neighbor node of node i
v^{ori}	The original number of valid nodes
v^{re}	The reduced number of valid nodes
v_k^{ch}	The number of child nodes of the k-th node in these v^{re} nodes
s_k^{ch}	The number of starting active slots of v_k^{ch}
p	The number of slots in which the active slots coverage of two forwarding nodes coincide
q_1, q_2	The number of sleep slots between active slots
γ	The number of child nodes
$t_{i,\gamma}^a$	The number of the active slots in $T_{\{1..\gamma\}}^c$ before partition
$t_{i,\gamma}^s$	$N - t_{i,\gamma}^a$, The number of the sleep slots not in $T_{\{1..\gamma\}}^c$ before partition
$t_{i,\gamma,j}^s$	The length of the j-th segment in $t_{i,\gamma}^s$
β	The number of active slots segments divided by the active slots
ξ_k	The length of the k-th in the β segments
λ	The number of discontinuous segments after the division
$t_{i,\gamma,\lambda}^a$	The number of the active slots in $T_{\{1..\gamma\}}^c$ after partition
$t_{i,\gamma,\lambda}^s$	$N - t_{i,\gamma,\lambda}^a$, The number of the active slots in $T_{\{1..\gamma\}}^c$ after partition

TABLE 2. *(Continued.)* Notations used in the paper.

$t_{i,\gamma,\lambda,j}^s$	The length of the j-th sleep slot in $t_{i,\gamma,\lambda}^a$ after partition
μ_k	The length of the k-th in the β segments after partition
r	the number of routing tables
Δr	The reduction of the routing table length
$min_q^{q_2}$	The minimum between q and q_2
$max_q^{q_2}$	The maximum between q and q_2

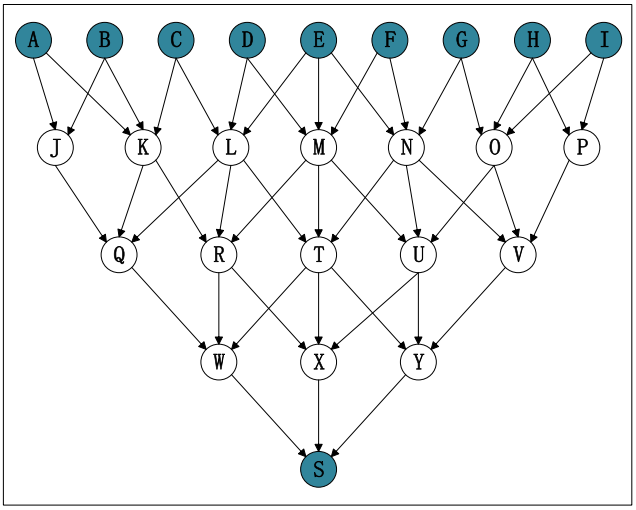


FIGURE 2. Network topology.

The nodes use the awake/sleep periodic rotation mode. N represents the number of time slots in a cycle, and number each slot with $\{1, 2, \ldots, N-1, N\}$. Each node selects part of the slot as the work slot, that is, the node is awake at these slots and sleep at other slots. M represents the number of work slots of nodes in one period ($M \leq N$). Figure 3 shows the slot state of each node in the network shown in Figure 2. In Figure 3, $N = 20$ and $M = 5$. In the typical sensor network, the node selects a starting time slot, and the active slots consist of its subsequent M consecutive active slots. As shown in Figure 3, the starting active slot of node A is slot 5, so the active slots are slots 5-9, and the sleep slots are 1-4 and 10-20.

The one-hop delay consists of the transmission delay and the wait delay. The wait delay can be expressed by the latency, and the transmission delay is always 1. Since the nodes are not always awake, the delay for each slot may be different. Although node i is ready to send the packet at slot t, it does not mean that it can be sent immediately. Only there is a forwarding node is active, the data can be transmitted. Otherwise, data cannot be transmitted until the earliest node wakes up in forwarding nodes. Therefore, the calculation formula of delay is as shown in Eq. (3).

$$d_i^t = \left(min\left(s_{i,j}^t\right) - t + N + 1\right) mod\ N \tag{3}$$

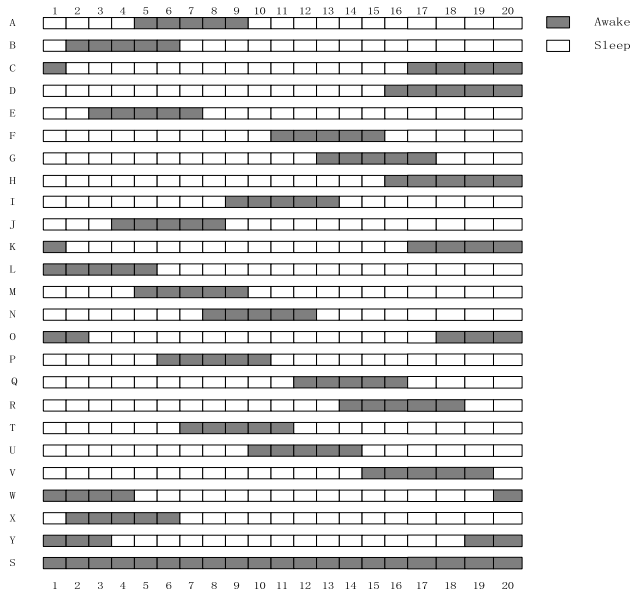


FIGURE 3. Duty cycle time slot.

where d_i^t indicates that the delay of node i at slot t, j represents one of the forwarding node of node i, $s_{i,j}^t$ indicates the time slot when the node j is awakened after slot t. and $\min(s_{i,j}^t)$ indicates the time slot when the earliest wake-up node is awakened in all forwarding nodes after slot t. The awakened slot of the earliest wake-up node minus the slot t is value of the latency.

For example, when node i wants to send packets at slot t, and there is a forwarding node is active at slot t, then the delay is 1 and the latency is 0. But if all forwarding nodes are sleep at slot t, and a node wakes up until slot $t+3$, another node wakes up at slot $t+4$. We can subtract slot t from the slot $t+3$ where the earliest node is awakened, so the delay is 4. What's more, if all forwarding nodes are sleep from slot t to slot N,and a node wakes up at slot $1(1 \leq t)$. Then add slot 1 to N to so that the result is greater than t, and use this result to minus t is the latency. Thus, the latency is $1+N-t$.

The following Eq. (4) gives the calculation formula of $\bar{d}_i$,which the value is the weighted average delay of all slots of this node.

$$\bar{d}_i = \left(\sum_{t=1}^{N} d_i^t\right) \Big/ N \tag{4}$$

The end to end delay $\mathcal{D}_{\mathrm{I}}$ refers to the average value of each one-hop delay on the route path when the packet routing of node i reaches the sink, as shown in Eq. (5).

$$\mathcal{D}_i = \sum \bar{d}_j | \; j \; is \; the \; node \; in \; the \; path \; of \; i \; to \; sink \tag{5}$$

According to the calculation method of the delay, Figure 4 gives the one-hop average delay of all nodes. There is a big gap in the average delay among nodes. The average one-hop average delay $\mathcal{D}_i$ is 3.75. In this case, the delay is large. The following three research motivations for improving network performance are given below.

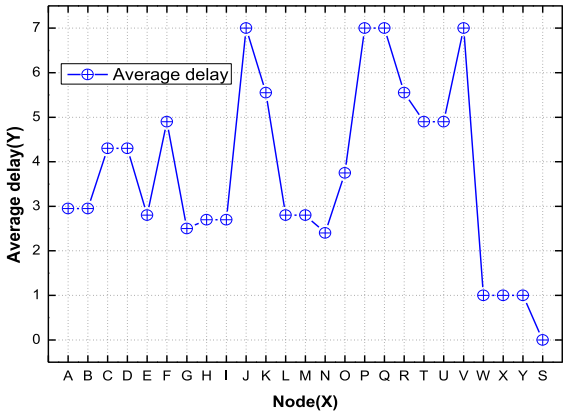


FIGURE 4. Average delay.

1) REDUCING LATENCY BY DIVIDING THE CONTINUOUS ACTIVE SLOTS INTO MULTIPLE SEGMENTS

The first research motivation is to divide continuous active slots into multi-segment small discontinuous active slots. Then there is no need to wait too long to have an active slot, thereby reducing latency. Figure 5 shows the slot state of each node that randomly divides continuous active slots into the discontinuous active slots. In such a slot division, the total number of active slots remains unchanged, so the energy consumption of the node remains the same.

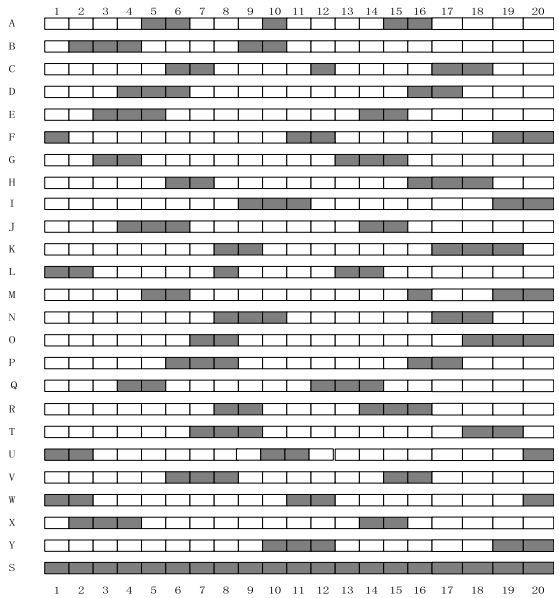


FIGURE 5. Duty cycle time slot with interval.

After dividing the continuous active slots into multiple active slots with smaller length, Figure 6 shows the comparison of the average delay between the continuous and discontinuous active slots. After the adjustment of the active slots, the delay of most nodes decreases significantly (only the delay of node G, H and I does not decrease). Overall, the average delay of the entire network is 2.324, and the

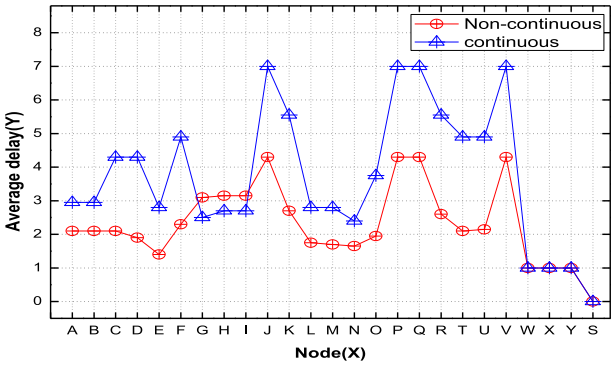


FIGURE 6. Average delay comparison between continuous and discontinuous.

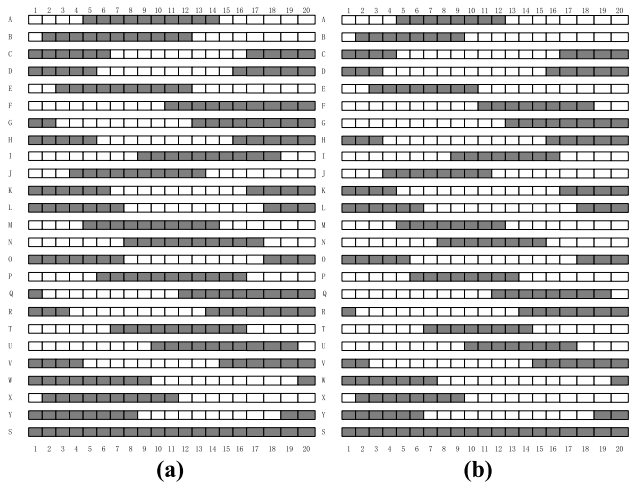


FIGURE 7. The slot state of node: (a) $M = 8$, (b) $M = 10$.

average latency is 1.324. By comparing the situation of the continuous active time slots, latency is reduced 1.426 slots, which reduces the latency by 51.9%. We can see that the scheme proposed to divide the continuous active time slot into multiple segments with smaller length can significantly reduce the latency.

2) REDUCING LATENCY BY UTILIZING THE RESIDUAL ENERGY IN THE FAR SINK REGION TO INCREASE ACTIVE TIME SLOTS

The main reason for the emergence of latency is that only M slots out of N slots in a cycle are in the active state. Therefore, when the sender has data to send, its forwarding nodes are sleep, which leads to latency. Obviously, adding active slots can reduce the latency. However, it will increase the energy consumption and reduce the lifetime meantime. Therefore, this method of reducing latency by consuming energy is hard to be used in wireless sensor networks. Besides, there is an imbalance in energy consumption in the wireless sensor networks. The nodes near the sink area are responsible for the forwarding of the data of the remote node, so the energy consumption is much higher than that in the remote sink area. In this way, the energy hole phenomenon will lead to the death of nodes within one hop of the sink and the death of the whole network in advance. According to the research results [38], when the network dies early due to the influence of the energy hole, there is still up to 90% more energy left in the network. Therefore, the second research motivation is to make full use of the residual energy of the nodes in the remote sink area and add active slots in these nodes.

The following experiment shows that increasing the value of M can effectively reduce the latency. In Figure 4, M is equal to 5. We increase M to 8 and 10 respectively, and then the slot state of the nodes is shown in Figure 7.

Figure 8 shows the comparison diagram of the average delay with different M. The black line means M is equal to 5, the blue line means M is equal to 8, and the red line means M is equal to 10. We can see from Figure 8 that, when M is 5, the average delay is the maximum. When M is 8, the average delay of each node is significantly reduced, and the improvement effect is significant. Although the average delay is already small, increasing M to 10 and the average delay can be further reduced. These experimental data indicating that the latency can be reduced by increasing M.

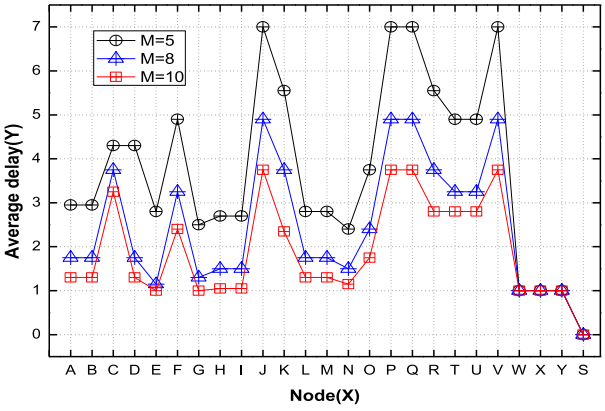


FIGURE 8. Average delay comparison with different M.

From the perspective of the entire network, when M is 5, the average delay is 3.75. And when M is 8, the average delay is 2.468, which is 1.282 slots lower than that when M is 5, reducing 34%. What's more, when M is 10, the average delay is 1.916, which is 1.834 lower than that when M is 5 and reduced by 49%. Meanwhile compared with the average delay when M is 8, the average delay decreased by 0.552 slots and reduced by 22%. This shows that increasing the number of active slots M can effectively reduce the latency. However, increasing the value of M will increase the energy consumption, which makes the previous research dare not adopt this method. It is found that there is a large amount of residual energy in the remote sink region. Thus, it can be fully utilized to increase M, effectively reducing latency without shorten lifetime.

In the network as Figure 2, the original strategy is setting M is equal to 5 for all nodes. In accordance with the residual

energy, adding the active slots of the nodes in the third and fourth layer to make M is equal to 10, so the slot arrangement of each node is as Figure 9. The comparison between the average delay and the original strategy of each node in the network is shown in Figure 10.

At this moment, the average delay is 3.092 after increased M to 10 of the nodes in the far sink region. Compared with the original strategy that the value of M is equal to 5, the average delay is reduced by 0.658 slots, which reduces by 18%. The scheme of adding active slots can significantly reduce the latency by utilizing residual energy.

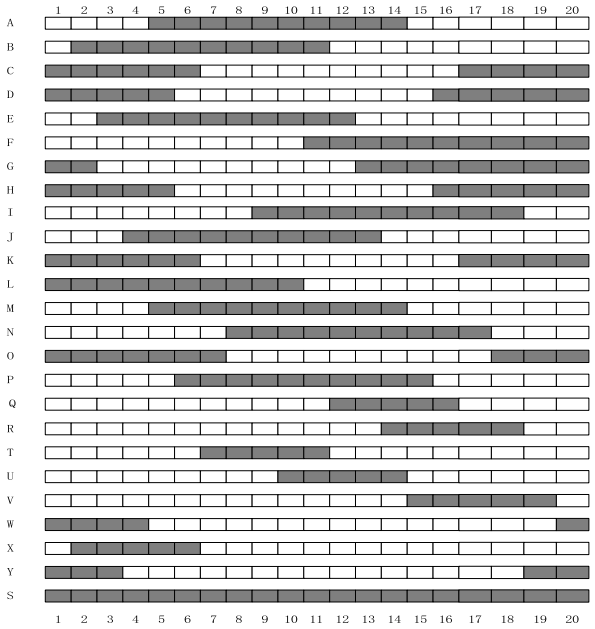


FIGURE 9. Increase M of the far sink area nodes.

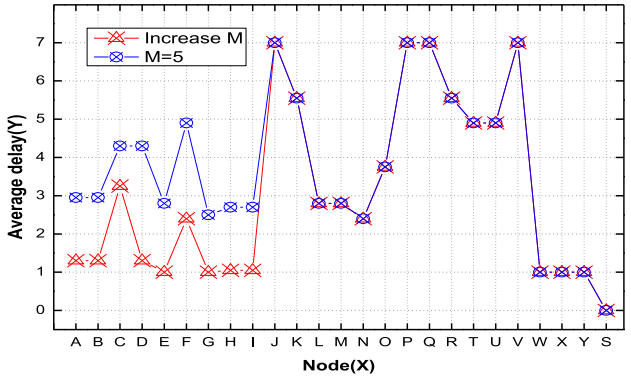


FIGURE 10. Average delay comparison after increasing M.

3) REDUCING THE NUMBER OF CANDIDATE RELAY NODES TO REDUCE THE ROUTING TABLE LENGTH

In the previous study, the main concern is on how to reduce the latency, rather than the routing table length. After the continuous active slots are divided into multiple segments, the routing table becomes longer. Reducing the routing table length can increase the speed of routing lookup and save the storage space of the routing table. Due to the small size, simple structure, limited energy and battery-driven characteristics of sensor nodes, it is a challenging problem to save the storage space of routing table as much as possible. Previous studies rarely achieved the problem of optimizing the delay and routing table simultaneously in duty cycle wireless sensor network. After increased M, it is possible to merge some segmented active slots into one segment, thereby reducing the routing table length as well as latency.

The third improvement strategy is to reduce the number of candidate relay nodes. It is found that when M is large enough, there are multiple forwarding nodes that may be selected as the relay node. At this time, if some forwarding nodes are designed to do not participate in the routing, the length of the routing table can be reduced effectively. These randomly selected nodes need to be far away from sink node, because the closer to sink, the more data it tends to undertake. In this case, removing the node close to sink may have a significant impact on the delay. How many nodes are selected, and which ones are selected are uncertain, balancing the delay and the routing table length.

Similarly, we use specific examples to illustrate the problem. When M is equal to 8, four nodes are randomly selected from the outermost and sub outermost layer of the network topology in Figure 2. Here nodes A, I, J, and P are excluded from the candidate relay nodes when routing. In this case, we let the four nodes that are subtracted look different from the other nodes, and the network topology diagram excluding the four nodes is shown in Figure 11.

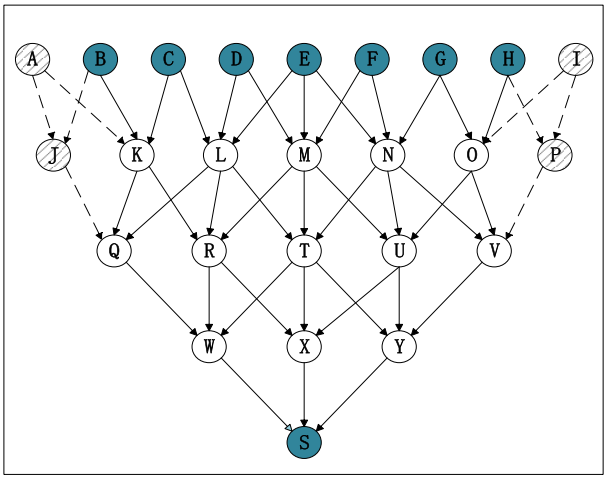


FIGURE 11. Network topology after reducing candidate relay nodes.

Here, the routing table design and storage mechanism refers to the concept of Non-increasing Delivery-latency Interval (NDI) of According [38]. The NDI-based routing scheme is to find the shortest delay in each NDI instead of each slot. While ensuring the minimum delay, it minimizes the storage of routing table (i.e. the size of the routing table).

Figure 1 shows the storage structure of the routing table. It consists of four items: starting slot, starting delay, ending delay, next hop.

Table 2(a) presents the simplified routing table of the six nodes A, B, H, I, J, P before limiting the number of candidate nodes. The starting slot and next hop are retained in the table, and the delay column is omitted. Table 2(b) represents a simplified routing table that excludes nodes A, I, J, and P. Compared with the previous strategy, this step reduced 12 routing table records.

TABLE 3. (a) Simplified original routing table. (b) Simplified routing table after reducing nodes..

(a)

Node	Starting slot	Next hop
A/B	1	K
	7	J
	14	K
H/I	1	O
	8	P
	17	O
J	1	Q
P	1	V

(b)

Node	Starting slot	Next hop
B	1	K
H	1	O

B. THE DESIGN OF SAC SCHEME

Figure 1 presents the storage structure of the Routing table (RT), The storage structure of routing broadcast packet (RBP) consists of three items: j, $s_j[\,]$, $e_j[\,]$. The symbol representation and meaning are as follows:

1. RT : routing table
2. RBP : routing broadcast packet, it is used to inform its neighbor nodes of its active slots state.
3. $startT$: starting slot
4. $startD$: starting delay
5. $endD$: ending delay
6. $nextH$: next hop
7. j : the node id
8. $s_j[\,]$: the starting slot array of node j
9. $e_j[\,]$: the ending slot array of node j

The routing broadcast packet of the node is used to inform its neighbor nodes of its active slots state. Since the active slots may be continuous or discontinuous, the slot state can be represented by an array of the beginning slot and the ending slot of each segment. That is to say, the route broadcast packet includes the node v_j, the start slot array $s_j[\,]$ and the end slot array $e_j[\,]$ of each segment of the active slots.

During the route initialization phase, all sensor nodes are awake. The routing table is empty and is built on each node in a distributed manner. First, the sink node broadcasts the routing broadcast packet. Then processing the routing broadcast packet. When there is no routing broadcast packet transmission in the network, the algorithm stops.

The processing procedure of the routing broadcast packet at node i can be described by Algorithm 1.

Algorithm 1 Processing RBP_j at Node i

1: $RT_{i,j} \leftarrow$BuildRT(RBP_j)
2: **If** RT_i is empty **then**
3: $\quad RT_i \leftarrow RT_{i,j}$
4: **Else**
5: $\quad T_{i,j} \leftarrow$GetT($RT_{i,j}$)
6: $\quad T_i \leftarrow$GetT(RT_i)
7: $\quad T_i \leftarrow T_{i,j} + T_i$
8: $\quad$ Sort T_i in ascending order
9: $\quad RT \leftarrow$null
10: $\quad$ **For** each T_i^k in T_i **do**
11: $\quad\quad$ Update (RT, RT_i, $RT_{i,j}$, T_i^k, $\&k$)
12: $\quad$ **End for**
13: $\quad RT_i \leftarrow RT$
14: **End if**
15: Broadcast RBP_i to its neighbors

The symbol representation and meaning in the algorithm are as follows:

1. RBP_j : the routing broadcast packet from node j
2. RT_i : the routing table of node i
3. $RT_{i,j}$: the route from node i to j
4. T_i : the starting slots set in RT_i
5. $T_{i,j}$: the starting slots set in $RT_{i,j}$
6. T_i^k : the k-th element in T_i

In Algorithm 1, node i first construct the route $RT_{i,j}$ based on RBP_j (line 1). If the RT_i is empty, then use $RT_{i,j}$ as RT_i (lines 2-3). Otherwise, to get the start slot of each NDI and join it to $T_{i,j}$ based on $RT_{i,j}$ (line 5). Then build T_i based on RT_i and merge $RT_{i,j}$ into RT_i (lines 6-7). Sort the starting slot in T_i in ascending order (line 8). Then an empty RT is built to store temporary routes during route updates (line 9). For each slot in T_i, update the current RT at slot T_i^k based on $RT_{i,j}$ and RT_i (lines 10-12). After that the updated RT is assigned to RT_i (line 13). Finally, node i broadcasts RBP_i to its neighbor nodes (line 15).

The process of constructing $RT_{i,j}$ can be described by Algorithm 2.

The symbol representation and meaning in the algorithm are as follows:

1. e_j^k : the k-th element in $e_j[\,]$

Since the first NDI always starts with slot 1, slot 1 is first added to T_j (line 1). Then traverse the ending slot of node j. Add e_j^k+1 to T_j as long as e_j^k is not the last time slot (lines 2-8). For each starting slot in T_j, the routing is calculated as follows: the starting delay is the delay of the starting slot. The ending delay is always 1, except that the ending delay of the last NDI is calculated as the delay of the last time slot (lines 9-18).

Algorithm 2 Building $RT_{i,j}$ From RBP_j

1: Insert(T_j, 1)
2: **For each** e_j^k in e_j [] **do**
3: **If** $e_j^k = N$ **then**
4: **break**
5: **Else**
6: Insert(T_j, $e_j^k + 1$)
7: **End if**
8: **End for**
9: **For each** e_j^k in T_j **do**
10: $RT_{i,j}^k.startT = e_j^k$
11: $RT_{i,j}^k.startL = \mathrm{d}_{i,j}^{e_j^k}$
12: $RT_{i,j}^k.nextH = j$
13: **If** $k = \mathrm{length}(T_j) - 1$ **then**
14: $RT_{i,j}^k.endL = d_{i,j}^N$
15: **Else**
16: $RT_{i,j}^k.endL = 1$
17: **End if**
18: **End for**

The process of obtaining $T_i$based on RT_ican be described by Algorithm 3. Simply traverse each routing record in RT_i and add the starting slot of NDI to the set T_i (lines 1-3).

Algorithm 3 Getting T_i From RT_i

1: **For each** RT_i^k in RT_i **do**
2: Insert(T_i, $RT_i^k.startT$)
3: **End for**

The process of updating RT at T_i^k based on $RT_{i,j}$ and RT_i can be described by Algorithm 4. In Algorithm 4, the delay at slot T_i^k with relay node is j and in the path of RT_i are firstly compared. If the delay is smaller in the case of relay node being j, RT will be updated with routing record of the slot T_i^k in $RT_{i,j}$ (lines 1-2). Only when the following two conditions are satisfied simultaneously, the value of k is increased by 1. It not only needs to satisfy the requirement of relay node for j with a smaller delay at slot T_i^{k+1}, but also need to satisfy the condition that the delay at slot T_i^{k+1} is less than or equal to the delay at the previous slot $T_i^{k+1} - 1$. Where the second condition is to guarantee that the delay is monotonically continuous and non-increasing within the range of slot T_i^k to slot T_i^{k+1}. Only in this way the two NDIs of slot T_i^k and slot T_i^{k+1} can be merged. Next to traverse the next slot T_i^k until these two conditions are not met (lines 3-5). Conversely, if the delay of relay node for j is greater (line 6), update RT with the route record of slot T_i^k in RT_i (line 7). Then the delay at the next starting slot T_i^{k+1} calculated by $RT_{i,j}$ and RT_i is also compared. Similarly, only when at slot T_i^{k+1} the delay of relay node for j is still greater and the delay at slot T_i^{k+1} is not greater than the delay at slot $T_i^{k+1} - 1$ satisfies both conditions, increase the value of k by 1. Traverse the next slot T_i^k until one of the two conditions is not satisfied (lines 8-10). There is a third case where both delays are equal (line 11). In this case, it needs to compare the delay calculated at the next starting slot T_i^{k+1} by $RT_{i,j}$ and RT_i. Update RT with the route record at slot T_i^k,which in the routing table with smaller delay, and increase the k value by 1. It represents the combination of two starting slots of slot T_i^k and slot T_i^{k+1}, after which loop traversal is performed (lines 12-23). If both delays at slot T_i^{k+1} are equal again, the strategy of updating RT with the route record at slot T_i^k in RT_i is taken (lines 24-26).

Algorithm 4 Updating RT at an Interval

1: **If** $\mathrm{d}_i^{T_i^k} > \mathrm{d}_{i,j}^{T_i^k}$ **then**
2: UpdateRT (RT, $RT_{i,j}$, T_i^k)
3: **While** $\mathrm{d}_i^{T_i^{k+1}} > \mathrm{d}_{i,j}^{T_i^{k+1}}$ and $\mathrm{d}_{i,j}^{T_i^{k+1}-1} \geq \mathrm{d}_{i,j}^{T_i^{k+1}}$ **do**
4: $k++$
5: **End while**
6: **Else if** $\mathrm{d}_i^{T_i^k} < \mathrm{d}_{i,j}^{T_i^k}$ **then**
7: UpdateRT (RT, RT_i, T_i^k)
8: **While** $\mathrm{d}_i^{T_i^{k+1}} < \mathrm{d}_{i,j}^{T_i^{k+1}}$ and $\mathrm{d}_i^{T_i^{k+1}-1} \geq \mathrm{d}_i^{T_i^{k+1}}$ **do**
9: $k++$
10: **End while**
11: **Else**
12: **If** $\mathrm{d}_i^{T_i^{k+1}} > \mathrm{d}_{i,j}^{T_i^{k+1}}$ **then**
13: UpdateRT (RT, $RT_{i,j}$, T_i^k)
14: $k++$
15: **While** $\mathrm{d}_i^{T_i^k} > \mathrm{d}_{i,j}^{T_i^k}$ and $\mathrm{d}_{i,j}^{T_i^k-1} \geq \mathrm{d}_{i,j}^{T_i^k}$ **do**
16: $k++$
17: **End while**
18: **Else if** $\mathrm{d}_i^{T_i^{k+1}} < \mathrm{d}_{i,j}^{T_i^{k+1}}$ **then**
19: UpdateRT (RT, RT_i, T_i^k)
20: $k++$
21: **While** $\mathrm{d}_i^{T_i^k} < \mathrm{d}_{i,j}^{T_i^k}$ and $\mathrm{d}_i^{T_i^k-1} \geq \mathrm{d}_i^{T_i^k}$ **do**
22: $k++$
23: **End while**
24: **Else**
25: UpdateRT (RT, RT_i, T_i^k)
26: **End if**
27: **End if**

After the routing table is generated, an improved strategy is proposed to reduce the routing table length by reducing the number of candidate relay nodes. The processing process after receiving the v [] can be described by Algorithm 5.

The symbol representation and meaning in the algorithm are as follows:

1. v [] : the node identification array
2. i : the node id
3. v_i^{ne} : the neighbor node of node i
4. j : the neighbor node of node v_i^{ne}

Algorithm 5 Updating RT After Reducing Candidate Relay Nodes

Input: the reduced relay nodes $v[]$
Output: The updated RT_i
1: **If** i is in $v[]$ **then**
2: **For each** n_i in i's neighbors **do**
3: Delete i from v_i^{ne}'s neighbors
4: Set $RT_{v_i^{ne}}$ as empty
5: **For each** j in v_i^{ne}'s neighbors **do**
6: j send RBP_j to v_i^{ne}
7: Building $RT_{v_i^{ne}}$ from RBP_j at node v_i^{ne}
8: **End for**
9: **End for**
10: **End if**

In Algorithm 6, node i first determines whether i is in $v[]$ (line 1). If it is included, indicating that itself is one of the subtracted nodes, and this message needs to notify its neighbor nodes to update the routing table. For each neighbor node v_i^{ne}, the first step is to remove i from its neighbors (lines 2-3). The second step is to set the routing table of v_i^{ne} empty (line 4). Next, let each neighbor node j send RBP_j to v_i^{ne}. The process of generating or updating $RT_{v_i^{ne}}$ can be described by Algorithm 1 (lines 5-8). After the routing table of i's neighbor nodes are fully updated, finally the routing table of node i needs to be emptied.

The process of looking up the routing table can be described by Algorithm 6. First traverses each record in the routing table of node i (line 1). Compare slot t to the starting slot until t is smaller than it (line 2). When t is smaller than the starting slot, select the next hop item in the previous route record as the next hop. Meanwhile, the delay corresponding to slot t and the slot to send data are calculated (lines 3-5), then the loop traversal of the routing table is completed (line 6). Another case is that after traversing all the records, there is no starting slot larger than t. This indicates that slot t is between the last starting slot and the slot N. In this case, the next hop of the last route record is selected as the relay node, and the delay of slot t and the slot for sending the packet are calculated (lines 8-12).

V. PARAMETER OPTIMIZATION AND PERFORMANCE ANALYSIS

A. CALCULATIONS OF AVERAGE DELAY

Theorem 1: There are N time slots and M consecutive active slots in a period. As for node j, s_j represents the starting active slot, and the distribution of active slots satisfies Eq. (6):

$$T_j^c = \begin{cases} [s_j, s_j + M - 1], & s_j + M - 1 \le N \\ [s_j, N] \cup [1, s_j + M - 1 - N], & s_j + M - 1 > N \end{cases} \tag{6}$$

Proof: The active slot range is the M slots after the starting active slot. When the M slots are all distributed in the same period, that is, $s_j + M - 1 \le N$, the distribution of

Algorithm 6 Search for Routing at Slot t

Input: node i receive data at slot t
Output: the next hop at slot t
1: **For each** RT_i^k in RT_i **do**
2: **while** $t < RT_i^k.startT$ **do**
3: Select $RT_i^{k-1}.nextH$ as next hop at slot t
4: $d_i^t = \max\{RT_i^{k-1}.startL - (t - RT_i^{k-1}.startT), 1\}$
5: Forward data at slot $(t + d_i^t - 1 + N) \bmod N$
6: Break the for loop
7: **End while**
8: **If** RT_i^k is the last one in RT_i **then**
9: Select $RT_i^k.nextH$ as next hop at slot t
10: $d_i^t = \max\{RT_i^k.startL - (t - RT_i^k.startT), 1\}$
11: Forward data at slot $(t + d_i^t - 1 + N) \bmod N$
12: **End if**
13: **End for**

active slots is shown as

$$T_j^c = [s_j, s_j + M - 1].$$

Conversely, when the M slots are not distributed in the same period, that is, $s_j + M - 1 > N$, the distribution of the active slots is shown as

$$T_j^c = [s_j, N] \cup [1, s_j + M - 1 - N].$$

□

Theorem 2: The active slots distribution of node j is T_j^c and satisfies Eq. (6). If node i wants to send data to node j at slot t, the delay calculation formula is shown as Eq. (7):

$$d_{i,j}^t = \begin{cases} 1, & t \in T_j^c \\ (s_j - t + N + 1)\ \%N, & t \notin T_j^c \end{cases} \tag{7}$$

Proof: If node j happens to be awake at slot t, namely $t \in T_j^c$, then data can be sent immediately, and the delay is 1.

If the slot t is beyond the scope of the active slots distribution, namely $t \notin T_j^c$, it needs to wait for the arrival of the earliest active slot after slot t. The earliest active slot is s_j, and it can be divided into two cases. The first case is when $s_j + M - 1 \le N$, and $t \in [1, s_j - 1] \cup [s_j + M, N]$. When $t \in [1, s_j - 1]$, the delay is $s_j - t + 1$. Likewise, when $t \in [s_j + M, N]$, the delay is $s_j - t + N + 1$.

The second case is when $s_j + M - 1 > N$, then $t \in [s_j + M - N, s_j - 1]$, so the delay is $s_j - t + 1$. □

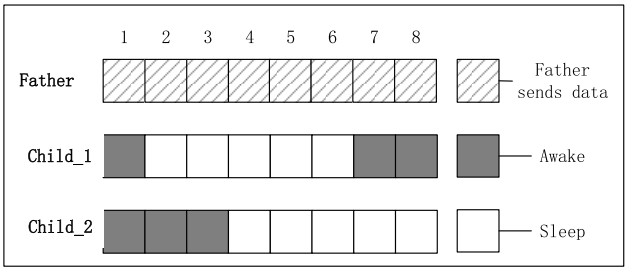


FIGURE 12. The example of theorem 1, theorem 2.

For example, Figure 12 shows an example of theorem 1 and theorem 2. As shown in Figure 12, $N = 8, M = 3$, there are one parent node and two child nodes. The first child node meets $s_1 = 7, T_1 = [7, 8] \cup [1]$; and the second child node meets $s_2 = 1, T_2 = [1, 3]$. According to Eq. (7), calculate the delay at each slot when there is only the first child node. Then calculate the delay of sending data at each slot when there are the first and second child nodes. The above two results are shown in Table 4. According to Eq. (3), when there are the first and second child nodes, the delay at each slot is shown in Table 5. According to Eq. (4), the weighted average delay of this node in all slots is 1.75.

TABLE 4. The calculation of theorem 1, theorem 2.

Slot	Delay		Latency		Slot to send	
	First	Second	First	Second	First	Second
1	1	1	0	0	1	1
2	6	1	5	0	7	2
3	5	1	4	0	7	3
4	4	6	3	5	7	1
5	3	5	2	4	7	1
6	2	4	1	3	7	1
7	1	3	0	2	7	1
8	1	2	0	1	8	1

TABLE 5. The calculation of Eq. (3).

Slot	Delay	Latency	Slot to send
1	1	0	1
2	1	0	2
3	1	0	3
4	4	3	7
5	3	2	7
6	2	1	7
7	1	0	7
8	1	0	8

To further elucidate the role of theorem one and theorem two, there is an example from network perspective. When the sender must send data at slot t, if there is at least one forwarding node is awake, so the one-hop delay is equal to 1. But if there is one awakening forwarding node until slot $t+j$, the one-hop delay is j.

Theorem 3: There are N time slots and M consecutive active slots in a period. Node i has γ child nodes, the active slots coverage of γ child-nodes has $t^a_{i,\gamma}$ time slots, whereas the rest of the $N - t^a_{i,\gamma}$ slots are divided into β segments by the active slots. Where the length of the k-th segment is ξ_k, then $\bar{d}_i$ is shown as Eq. (8):

$$\bar{d}_i = \frac{\left(t^a_{i,\gamma} + \sum_{k=1}^{\beta} \sum_{j=2}^{\xi_k+1} j\right)}{N} \tag{8}$$

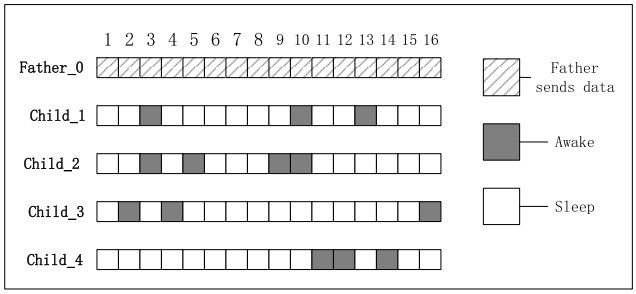
Proof: According to Eq. (4) and Eq. (7), the calculation process of delay is as

$$\begin{aligned}\bar{d}_i &= \left(\sum_{t=1}^{N} d_i^t\right) \Big/ N = \left(\sum_{t=1}^{N} \min_{j\varepsilon\{1..\gamma\}} d_{i,j}^t\right) \Big/ N \\ &= \frac{\left(t^a_{i,\gamma} + \overbrace{2+\ldots+\xi_1+1}^{\xi_1} \ldots \overbrace{2+\ldots+\xi_\beta+1}^{\xi_\beta}\right)}{N} \\ &= \frac{\left(t^a_{i,\gamma} + \sum_{k=1}^{\beta} \sum_{j=2}^{\xi_k+1} j\right)}{N}.\end{aligned}$$

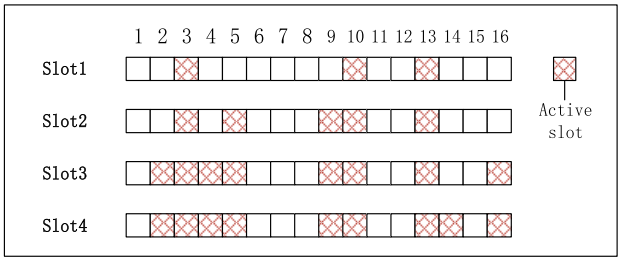
From the network perspective, first set an array with a length of a period. The initial value is all 0, indicating that they are all dormant. We use the array to record whether the slot in the number has a forwarding node in the awake state, and the value is set to 1. The array corresponds to the awaken sequence of the forwarding nodes. The number of values with one corresponds to the total number of active slots $t^a_{i,\gamma}$, and several consecutive zeros correspond to the sleep slots, and then Eq. (8) is used to calculate the delay.

For example, Figure 13 (a) shows the slot state of one parent node and four candidate relay nodes. Figure 13 (b) shows the process of calculating the delay. As shown in Figure 13, when there is only node 1 as the candidate relay node, the active slots coverage is as shown in Figure 13 (b) Slot 1. Only three slots numbered 3, 10, and 13 are in the active slot's coverage, namely 3, 10, $13 \in T_1^c$ and $t^a_{0,1} = 3$. The rest of the $N - t^a_{0,1} = 13$ slots are divided into $\beta = 3$ segments. The first segment is $[14, 16] \cup [1, 2]$ and the length is $\xi_1 = 5$. The second segment is $[4, 9]$ and the length is $\xi_2 = 6$. The third segment is $[11, 12]$ and the length is $\xi_3 = 2$. $\bar{d}_i = \frac{\left(t^a_{0,1}+\sum_{k=1}^{\beta}\sum_{j=2}^{\xi_k+1} j\right)}{N} = \frac{\left(3+\sum_{j=2}^{6} j+\sum_{j=2}^{7} j+\sum_{j=2}^{3} j\right)}{16} = 3.4375$.

As shown in Figure 13, when there is node 1 and node 2 as the candidate relay nodes, the active slots coverage is shown in Figure 13 (b) Slot 2. We can see only 5 slots numbered 3, 5, 9,10, 13 are in the active slots coverage, namely $3, 5, 9, 10, 13 \in T^c_{\{1,2\}}$ and $t^a_{0,2} = 5$. The rest of the $N - t^a_{0,2} = 11$ slots are divided into $\beta = 4$ segments. The first segment is $[14, 16] \cup [1, 2]$ and the length is $\xi_1 = 5$. The second segment is $[4]$ and the length is $\xi_2 = 1$. The third segment is $[6, 8]$ and the length is $\xi_3 = 3$. The last segment is $[11, 12]$ and the length is $\xi_4 = 2$. This change occurs due to child node 2 is added, and the original segment $[4, 9]$ is divided into two segments with an active slot numberd 5. The coverage range of active slots is widened, $\bar{d}_i = \frac{\left(5+\sum_{j=2}^{6} j+\sum_{j=2}^{2} j+\sum_{j=2}^{4} j+\sum_{j=2}^{3} j\right)}{16} = 2.5625$. Similarly, when the candidate relay nodes added node 3, its active slot's coverage is as shown in Slot 3. Since the child node 3 is added, the slots numbered 3 and 5 with the original interval are connected, and the slot numbered 16 divides the original sleep slots into two segments. At this time, $\bar{d}_i = \frac{\left(8+\sum_{j=2}^{2} j+\sum_{j=2}^{4} j+\sum_{j=2}^{3} j+\sum_{j=2}^{3} j\right)}{16} = 1.8125$. When node 4 is



(a)



(b)

FIGURE 13. The example of theorem 3: (a) duty cycle time slot, (b) active slots coverage.

added again, its active slots coverage is as shown in Slot 4, and $\bar{d}_i = \frac{\left(8+\sum_{j=2}^{2} j+\sum_{j=2}^{4} j+\sum_{j=2}^{3} j+\sum_{j=2}^{2} j\right)}{16} = 1.6875$. □

Theorem 4: There are N time slots and M consecutive active slots in a period, the average delay $\bar{d}_i$ decreases with the increase of M. Under the premise that the number of j is equal to 2, the number of slots in which the active slots coverage of two forwarding nodes coincide is set as p, $p \in [0, M]$. The calculation formula of the average delay is shown as Eq. (9):

$$\bar{d}_i = \begin{cases} \dfrac{\left(M^2+N^2-2MN+3N-M\right)}{2N}, & p = M \\ \dfrac{2q^2-2\left(N-2M\right)q+(N-2M)^2+3N-2M}{2N}, & p = 0 \\ \dfrac{((N-2M+p)^2+3N-2M+p)}{2N}, & p \in (0, M) \end{cases} \quad (9)$$

Proof: If the number of the candidate relay nodes is 1, the delay of node i within the active slots coverage of forwarding node j is 1, where the latency is 0. There are M slots with a delay of 1, and the rest $N - M$ slots are outside the active slots coverage of j. The delay is one of $\{2, 3, \ldots, N-M, N-M+1\}$. Based on Eq. (8), the average delay can be calculated as shown in Eq. (10) :

$$\begin{aligned} \bar{d}_i &= \left(\sum_{t=1}^{N} d_i^t\right) \Big/ N \\ &= \frac{\left(M+\sum_{j=2}^{N-M+1} j\right)}{N} - \frac{\left(M+\frac{(2+(N-M+1))(N-M)}{2}\right)}{N} \\ &= \frac{\left(M^2+N^2-2MN+3N-M\right)}{2N} \\ &= \frac{\left(M-\left(N+\frac{1}{2}\right)\right)^2}{2N} - \frac{1}{8N} + \frac{1}{2} \end{aligned} \quad (10)$$

Taking N in Eq. (10) as a constant, the relationship between $\bar{d}_i$ and M is analogous to the unary quadratic equation. Figure 14 shows the rough image of this relationship. The axis of symmetry of the curve is $x = N + \frac{1}{2}$, and the minimum value $y = \frac{1}{2} - \frac{1}{8N}$. The y value gradually increases from the axis of symmetry to both sides. $M \leq N$, so $\bar{d}_i$ decreases as M increases. In order to make $\bar{d}_i$ as small as possible, the value of M needs to be as close as possible to N. However, increasing the value of M will increase the energy consumption of the node, which makes the previous research dare not adopt this method.

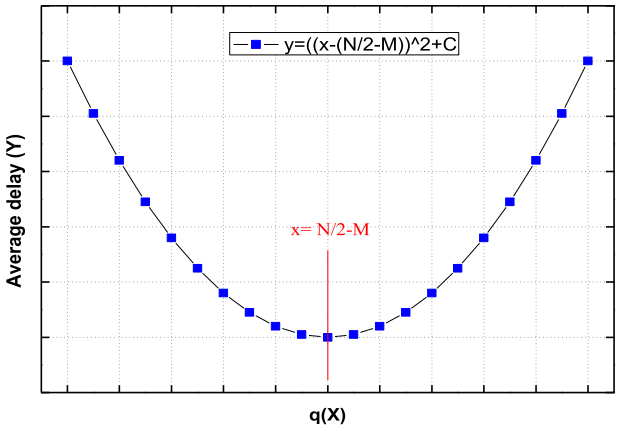


FIGURE 14. The relationship between M and $\bar{d}_i$.

If the number of candidate relay nodes does not exceed 2, the number is 1 can be regarded as a case of the number of being 2, which with the same range of the active slots. When the active slots of two candidate relay nodes overlaps, set the number of slots as p, $p \in [0, M]$. According to the value of p, it can be divided into three cases. In the first case, when $p = M$, it is equivalent to the case of one candidate relay node. The calculation formula of the average delay is as shown in Eq. (10). In the second case, when $p = 0$, it means that the active slots range of the two child nodes does not coincide. There are $2M$ slots with a delay of 1. Set the starting active slot of two child nodes as s_1, s_2 $(s_1 < s_2)$. Then, two segments of active slots are separated by $(s_2 - s_1 - M)$ and $(N - 2M - (s_2 - s_1 - M))$ sleep time slots. Let $q = s_2 - s_1 - M, q_2 = N - 2M - (s_2 - s_1 - M)$, and $q + q_2 = N - 2M$. The delays outside the active slot coverage are the values in $\{2, 3, \ldots, q+1\}$ and $\{2, 3, \ldots, q_2+1\}$, respectively. According to Eq. (8), the average delay can be calculated as shown in Eq. (11):

$$\begin{aligned} \bar{d}_i &= \left(\sum_{t=1}^{N} d_i^t\right) \Big/ N \\ &= \frac{\left(2M+\sum_{k=2}^{q+1} k+\sum_{k=2}^{q_2+1} k\right)}{N} \\ &= \frac{2q^2-2\left(N-2M\right)q+(N-2M)^2+3N-2M}{2N} \end{aligned} \quad (11)$$

It is known from Eq. (11) that $\bar{d}_i$ is determined by three physical quantities: the number of slots in the period N, the number of active slots M, and the interval slots between the active slots q. In general, N is a constant in the network; q and M are variables. Therefore, we can divide it into two cases: when M is regarded as a constant, q is a variable, and when both q and M are variables to discuss separately.

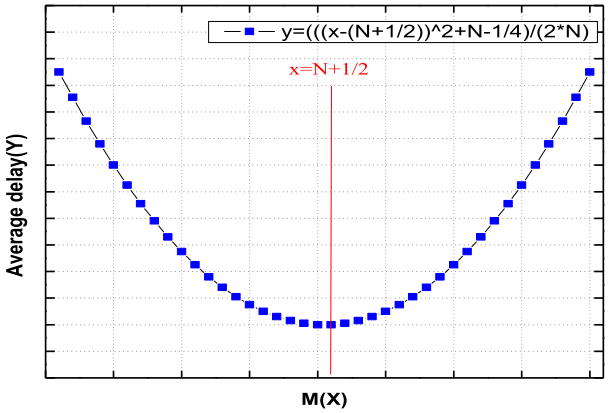


FIGURE 15. The relationship between q and $\bar{d}_i$.

In the case where N and M are regarded as constants and q is a variable, the relationship between $\bar{d}_i$ and q is analogous to a unary quadratic equation. Figure 15 shows the rough image of this relationship. The axis of symmetry of the curve is $x = \frac{N}{2} - M$, and the minimum value is $y = \frac{(\mathrm{N}-2\mathrm{M})^2}{2} + 3\,(\mathrm{N} - 2\mathrm{M}) + 4M$ at the axis of symmetry. The y value gradually increases from the axis of symmetry to both sides. $q \in [0, N - 2M]$, thus, when $s_2 - s_1 = \frac{N}{2}$, that is to say, when two intervals of the two active slots range are equal, $\bar{d}_i$ is the minimum. In the case of generalization to multiple child nodes, the analogy can be obtained that $\bar{d}_i$ reaches the minimum value when the active slots range of multiple child nodes is evenly distributed.

In the other case where N is regarded as a constant, q and M are variables, Figure 16 shows four curves of $\bar{d}_i$ varies with M when q takes different values. As shown in Figure 16, when q is a constant, $\bar{d}_i$ decreases along with the increase of M on the same curve and gradually approaches 1. The decreasing rate is lower and lower, and the curve is flatter and flatter. When M is a constant, the points on the four curves are compared vertically in parallel with the y-axis direction. When $M < 7$, $\bar{d}_i$ decreases as q increases, but when $M > 7$, this trend is not satisfied. Combined with the above discussion that M is a constant, it can be concluded that $\bar{d}_i$ decreases as q approaches $\frac{N}{2} - M$, and increases as q moves away from $\frac{N}{2} - M$.

In the third case, when $p \in (0, \mathrm{M})$, that is to say, there is an overlap of p slots in the active slots range of two child nodes. At this time, there are $2M - p$ slots with a delay of 1, and the delay outside the active slots range is the value in $\{2, 3, \ldots, N - 2M + p + 1\}$. According to Eq. (8),

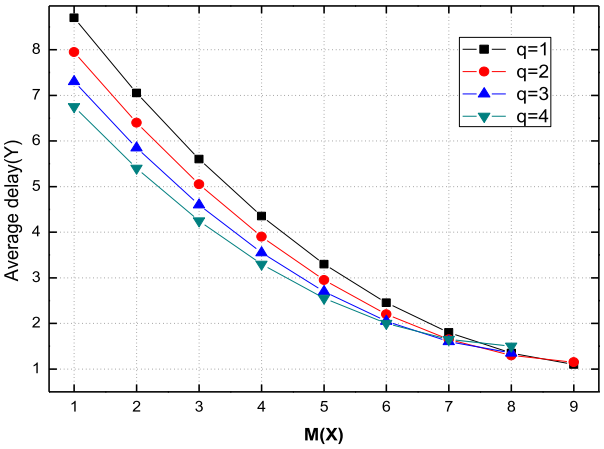


FIGURE 16. The relationship between q, M and $\bar{d}_i (N = 20)$.

the average delay can be calculated as shown in Eq. (12) :

$$\begin{aligned} \bar{d}_i &= \left(\sum_{t=1}^{N} d_i^t\right) \Big/ N \\ &= \frac{\left(2M - p + \sum_{k=2}^{N-2M+p+1} k\right)}{N} \\ &= \frac{(\mathrm{N}-2\mathrm{M}+p)^2 + 3N - 2M + p}{2N} \end{aligned} \tag{12}$$

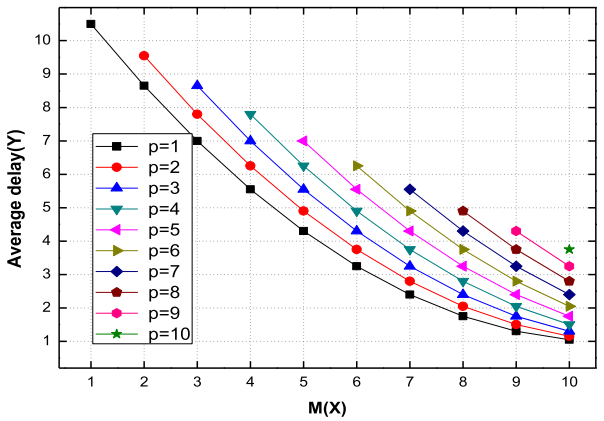


FIGURE 17. The relationship between p, M and $\bar{d}_i$.

In the case where N as a constant, p and M are variables in Eq. (12), Figure 17 shows ten curves $\bar{d}_i$ varies with M when p takes different values. As shown in Figure 17, when p is constant, $\bar{d}_i$ decreases along with the increase of M on the same curve and gradually decreases to close to 1. The decreasing rate is lower and lower, and the curve becomes more and more gradual. When M is constant, the points on the ten curves are compared vertically in parallel with the y-axis direction. It's not hard to find that $\bar{d}_i$ decreases as p increases, and the increase rate also will decrease. □

Theorem 5: The average delay of node i is calculated as Eq. (8). At this time, M is increased to $M+r$, obviously $r > 0$ and $M + r < N$. According to the values of p and r, the

formula for calculating $\Delta\bar{d}_i$ is as shown in Eq. (13).

$$\Delta\bar{d}_i = \begin{cases} \dfrac{(2N-2M+1-r)\,r}{2N}, & condition\ 1 \\ \dfrac{(2N-2M+1-r)\,r}{N}, & condition\ 2 \\ \dfrac{2\left(min_q^{q_2}\right)^2 + 2min_q^{q_2} r - r^2 + 1}{2N}, & condition\ 3 \\ \dfrac{q\,(q+1) + q_2\,(q_2+1)}{2N}, & condition\ 4 \\ \dfrac{(2\,(N-2M+p) - r + 1)\,r}{2N}, & condition\ 5 \end{cases} \tag{13}$$

In Eq. (13), the corresponding conditions are as follows: condition 1 is $p = M$, condition 2 is $p = 0$ and $r \le min_q^{q_2}$, condition 3 is $p = 0$ and $min_q^{q_2} < r < max_q^{q_2}$, condition 4 is $p = 0$ and $r \ge max_q^{q_2}$, and the last condition 5 is $p \in (0, M)$. And $\Delta\bar{d}_i$ is the symbol of reduced average delay, p is the number of slots in which the active slots ranges of the two forwarding nodes overlap, q and q_2 are the number of sleep slots respectively separated by active slots, $min_q^{q_2}$ and $max_q^{q_2}$ represent the smaller value and the lagrger value of q and q_2.

Proof: Here, according to the value of p, it can be divided into the following five cases. In the first case, when $p = M$, M is increased to $M + r$, and the calculation formula of reduced average delay can be shown as Eq. (14) :

$$\begin{aligned} \Delta\bar{d}_i &= \frac{M^2 + N^2 - 2MN + 3N - M}{2N} \\ &\quad - \frac{(M+r)^2 + N^2 - 2\,(M+r)\,N + 3N - (M+r)}{2N} \\ &= \frac{(2N - 2M + 1 - r)\,r}{2N} \end{aligned} \tag{14}$$

In the case where N as a constant, r and M are variables in Eq. (14), Figure 18 shows five curves of $\Delta\bar{d}_i$ varies with r when M takes different values. When M is constant, $\Delta\bar{d}_i$ increases with the increase of r on the same curve, but the rate of decrease is smaller and smaller. When r is equal, the points on the curves are compared vertically in parallel with the y-axis direction. $\Delta\bar{d}_i$ decreases as M increases, indicating that the smaller the value of M, the more the average delay is reduced when the same r is increased.

In the second case, when $p = 0$, increase M to $M + r$, then the value of r will appear in three cases. The first case is when $r \le min_q^{q_2}$. The second case is when $min_q^{q_2} < r < max_q^{q_2}$. The third case is when $max_q^{q_2} \le r$. The following are based on these three cases to calculate $\Delta\bar{d}_i$.

a. When $r \le min_q^{q_2}$, there is still no overlap in the active slots range of the two forwarding nodes. That is to say, $T_1^c \cap T_2^c = \emptyset$. There are $2\,(M + r)$ slots with a delay of 1, and the two segments active slots are separated by $(q - r)$ and $(q_2 - r)$ time slots respectively. The calculation formula of reduced average delay as shown in Eq. (15).

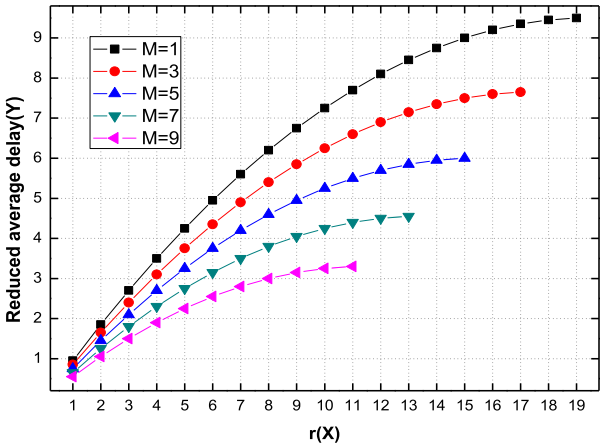


FIGURE 18. The relationship between r, M and $\Delta\bar{d}_i$.

$$\begin{aligned} \Delta\bar{d}_i &= \frac{\left(2M + \sum_{k=2}^{q+1} k + \sum_{k=2}^{q_2+1} k\right)}{N} \\ &\quad - \frac{\left(2\,(M+r) + \sum_{k=2}^{q-r+1} k + \sum_{k=2}^{q_2-r+1} k\right)}{N} \\ &= \frac{(2N - 2M + 1 - r)\,r}{N} \end{aligned} \tag{15}$$

In this case, $\Delta\bar{d}_i$ is exactly twice as much as when $p = M$. Thus the relationship between $\Delta\bar{d}_i$, M and r is similar to Figure 18, except that the ordinate $\Delta\bar{d}_i$ is doubled.

b. When $min_q^{q_2} < r < max_q^{q_2}$, there is an overlap in the active slots range of two forwarding nodes, that is, $T_1^c \cap T_2^c \ne \emptyset$. There are $2M + r + min_q^{q_2}$ slots with a delay of 1, and $\left(max_q^{q_2} - r\right)$ sleep slots. The calculation formula of reduced average delay as shown in Eq. (16), as shown at bottom of the next page.

Eq. (16) can be divided into two cases to discuss separately: N and M as constants, r and q as variables, and N, q as constants, r and M as variables. In the case of r and q as variables, let $N = 20$, $M = 3$, Figure 19 shows five curves of $\Delta\bar{d}_i$ varies with r when q takes different values.

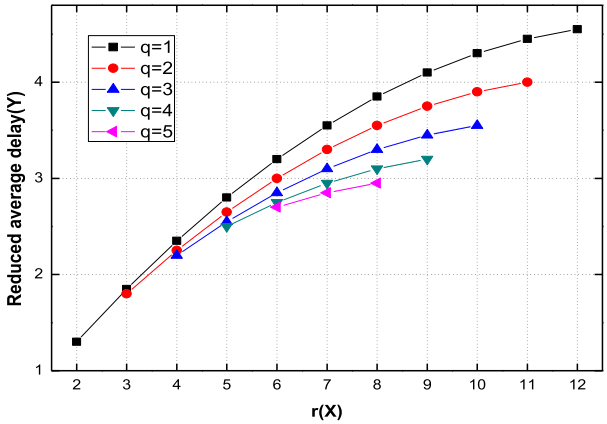


FIGURE 19. The relationship between r, q and $\Delta\bar{d}_i$.

As shown in Figure 19, when q is a constant, $\Delta\bar{d}_i$ increases with the increase of r on the same curve and the increase rate gradually decreases. When r is equal, the points on the curves are compared vertically in parallel with the y-axis direction. The comparison shows that when the same r value is increased, $\Delta\bar{d}_i$ decreases as q increases.

In another case, when r and M as variables, let $N = 20$, $q = 1$. Figure 20 shows four curves of reduced average delay $\Delta\bar{d}_i$ varies with r when M takes different values. As shown in Figure 20, when M is constant, $\Delta\bar{d}_i$ increases with the increase of r on the same curve and the rate of increase gradually decreases. When r is equal, the points on the curves are compared vertically in parallel with the y-axis direction. It also shows that $\Delta\bar{d}_i$ decreases as M increases, indicating that the smaller the M value, the more the average delay is reduced when the same r value is increased.

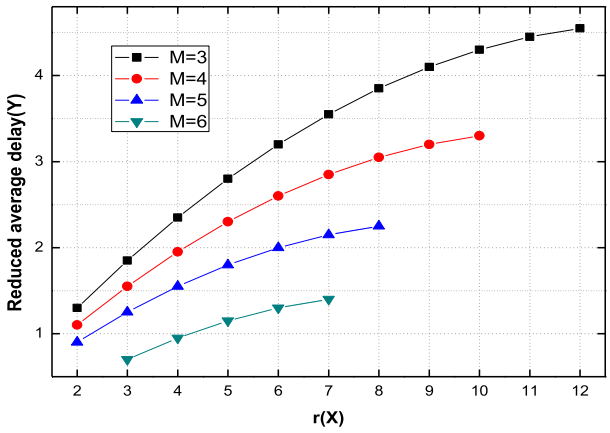


FIGURE 20. The relationship between r, M and $\Delta\bar{d}_i$.

c. When $max_q^{q_2} \leq r$, active time slot range has fully covered the entire period, that is, $T_1^c \cup T_2^c = \{1 \ldots 20\}$. There are N time slots with a delay of 1, so the average delay is 1. The calculation formula of reduced average delay is as shown in Eq. (17):

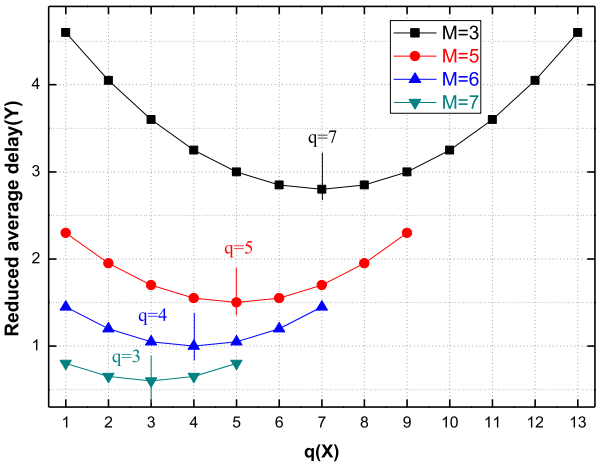


FIGURE 21. The relationship between q, M and $\Delta\bar{d}_i$.

$$\begin{aligned}\Delta\bar{d}_i &= \frac{\left(2M + \sum_{k=2}^{q+1} k + \sum_{k=2}^{q_2+1} k\right)}{N} - 1 \\ &= \frac{2q^2 - 2(N-2M)q + (N-2M)^2 + 3N - 2M}{2N} - 1 \\ &= \frac{2q^2 - 2(N-2M)q + (N-2M)^2 + N - 2M}{2N} \\ &= \frac{q(q+1) + q_2(q_2+1)}{2N}\end{aligned} \tag{17}$$

In the case where N as a constant, q and M as variables in Eq. (17), Figure 21 gives four curves of $\Delta\bar{d}_i$ varies with q when M takes different values. As shown in Figure 21, when M is constant, the curve of $\Delta\bar{d}_i$ changing with q on the same curve is a parabola with an upward opening. Its axis of symmetry is $x = \frac{N}{2} - M$, which means that $\Delta\bar{d}_i$ gets the minimum value when the interval of dormancy slot q is equal to q_2.

In the third case, when $p \in (0, M)$, M is increased to $M+r$. There is an overlap of $p + r$ slots in the active slots range of two forwarding nodes. At this time, there are $2M - p + r$ slots with a delay of 1, and the delay outside the active slots range

$$\begin{aligned}\Delta\bar{d}_i &= \frac{\left(2M + \sum_{k=2}^{q+1} k + \sum_{k=2}^{q_2+1} k\right)}{N} - \frac{\left(2M + r + min_q^{q_2} + \sum_{k=2}^{max_q^{q_2}-r+1} k\right)}{N} \\ &= \frac{2q^2 + 2(N-2M)(2r - |q-r| - q) - (2r - |q-r|)^2 + 2r - |q-r|}{2N} \\ &= \frac{2\left(min_q^{q_2}\right)^2 + 2min_q^{q_2} r - r^2 + 1}{2N} \\ &= \begin{cases} \dfrac{2q^2 + 2(N-2M)(3r-2q) - (3r-q)^2 + 3r - q}{2N}, & q > r \\ \dfrac{2q^2 + 2(N-2M)r - (r+q)^2 + r + q}{2N}, & q \leq r \end{cases}\end{aligned} \tag{16}$$

is the value in $\{2, 3, \ldots, N-2M+p-r+1\}$. The reduced average delay can be calculated as shown in Eq. (18) :

$$\Delta\bar{d}_i = \frac{\left(\sum_{k=2}^{N-2M+p+1} k - r - \sum_{k=2}^{N-2M+p-r+1} k\right)}{N} = \frac{(2(N-2M+p)-r+1)\,r}{2N} \tag{18}$$

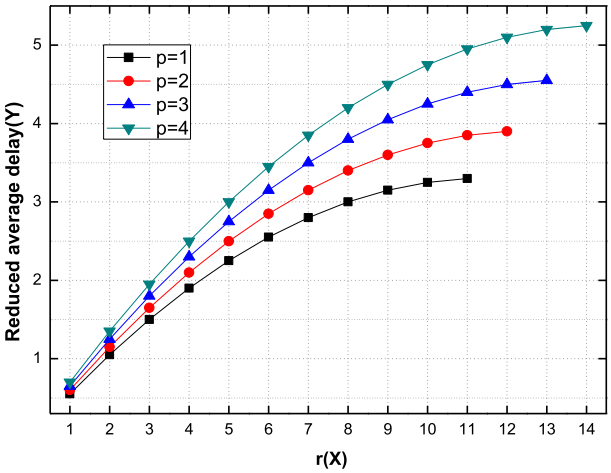


FIGURE 22. The relationship between p, r and $\Delta\bar{d}_i$.

When N and M are regarded as variables, r and p are variables, Figure 22 shows four curves of $\Delta\bar{d}_i$ varies with r when p takes different values. As shown in Figure 22, when p is constant, $\Delta\bar{d}_i$ increases with the increase of r on the same curve and the increase rate gradually decreases. When r is the same, the points on the curves are compared vertically in parallel with the y-axis direction. It shows that $\Delta\bar{d}_i$ increases as p increases, indicating that the average delay decreases with the increase of the number of active slots, but the effect of reduction is less and less obvious.

Above all, Eq. (14), Eq. (15), Eq. (16), Eq. (17), and Eq. (18) respectively give the calculation formulas of $\Delta\bar{d}_i$under five circumstances, so the calculation formula of Eq. (13) can be obtained. □

Theorem 6: There are N time slots and M consecutive active slots in a period. If M is increased to $M+r$, under the premise of theorem 5, the calculation formula of ΔD_i at best, worst and general conditions is as shown in Eq. (19), as shown at the bottom of the next page.

Proof: Although the starting active slot of nodes is random, when the starting active slot s_1 of the first child node is determined, and the distribution of the active slots is shown as $T_1^c = [s_1, s_1+M-1]$ or $T_1^c = [s_1, N] \cup [1, M+s_1-1-N]$. The selection of the second child node s_2 will directly affect the probability of p in different ranges. $p \in [0, M]$ is divided into three pieces. When $p = M$, it also satisfies: $T_1 = T_2$ and $s_2 = s_1$, the probability of this happening is $\frac{1}{C_N^1}$. When $p = 0$, that is, $T_1^c \cap T_2^c = \emptyset$. It is equivalent to selecting a slot from $N-2M$ time slots as s_2. The probability of this happening is $\frac{C_{N-2M}^1}{C_N^1}$. At last, the probability of the occurrence of $p \in (0, \mathrm{M})$ is $1 - \frac{1}{C_N^1} - \frac{C_{N-2M}^1}{C_N^1}$.

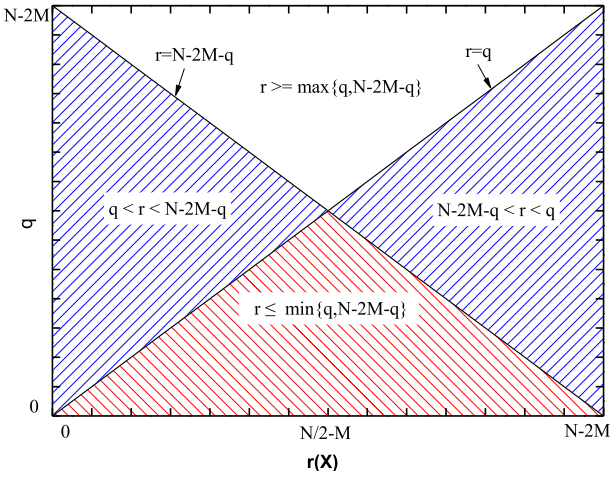


FIGURE 23. The probability of r and q.

In the case of $p = 0$, set q as the number of sleep slots at the interval between the active slots range of two nodes. Obviously, $q \in [0, N-2M]$, where each value of q in the range is an equal probability. Since $r \in [0, N-2M]$, the value of r in this range is also an equal probability. Figure 23 represents the relationship between q and r. As shown in Figure 23, the equations of the two diagonal lines are $r = q_2$ and $r = q$ respectively. It divides a square area with side length of $N-2M$ into four blocks, where the blue shaded area on both sides satisfies $min_q^{q_2} < r < max_q^{q_2}$, the red shaded area at the bottom satisfies $r \le min_q^{q_2}$, and the blank area at the top satisfies $r \ge max_q^{q_2}$. The area of each area represents the probability that the values of q and r meet the condition of the area, so the probabilities of the three cases are $\frac{1}{2}$, $\frac{1}{4}$ and $\frac{1}{4}$ respectively.

In the general case, the probability of $p = M$, $p = 0$ and $p \in (0, \mathrm{M})$ is $\frac{1}{N}$, $\frac{N-2M}{N}$ and $\frac{2M-1}{N}$ respectively. When $p = 0$, $\Delta\bar{d}_i$ depends on q. For the convenience of calculation, use $q = \frac{N}{2} - M$ for estimation. The weighted average is calculated according to the value of p. At this time, the calculation formula of $\Delta\mathcal{D}_i$ is shown as bottom of the next page.

In the best case, when the $\Delta\bar{d}_i$ of each node is larger, the overall $\Delta\mathcal{D}_\mathrm{I}$ is larger. All nodes satisfy the condition of $p = 0$, $q = 0$ or $q = N-2M$. The calculation formula of $\Delta\mathcal{D}_i$ is shown as

$$\Delta\mathcal{D}_i = \frac{(2N-2M+1-r)\,r}{N}.$$

In the worst case, the smaller $\Delta\bar{d}_i$ of each node, the smaller $\Delta\mathcal{D}_i$ of the whole. All nodes satisfy the condition of $p = M$. The calculation formula of $\Delta\mathcal{D}_i$ is just half of the best case, and is shown as

$$\Delta\mathcal{D}_i = \frac{(2N-2M+1-r)\,r}{2N}.$$

□

Table 6 gives the value of $\Delta\mathcal{D}_i$ after increasing M to $M+r$ with different values of r when $N = 20$ and $M = 5$. The three columns from left to right are the value of $\Delta\mathcal{D}_i$ in the general case, best case and worst case respectively.

TABLE 6. The example of theorem 6.

r	Generally	Best	Worst
1	0.93	1.5	0.75
2	1.83	2.9	1.45
3	2.71	4.2	2.1
4	3.57	5.4	2.7
5	4.40	6.5	3.25

Theorem 7: There are N time slots and M consecutive active slots in a period, node i has γ forwarding node. Here the continuous active slot is divided into discontinuous active slots. There is only based on the case of $\gamma > 1$ to calculate $\Delta\bar{d}_i$. The calculation formula of $\Delta\bar{d}_i$ in the best, worst and general case as shown in Eq. (20), as shown at the bottom of the next page. where $t^a_{i,\gamma}$ is the number of time slots within the active slots coverage of the γ forwarding nodes before the partition, β is the number of segments in which $N - t^a_{i,\gamma}$ sleep slots are divided by the active slots. The length of the j-th segment is $t^s_{i,\gamma,j}|j\epsilon 1..\beta$ and it satisfies $\sum_{j=1}^{\beta} t^s_{i,\gamma,j} = N - t^a_{i,\gamma}$. Where $t^a_{i,\gamma,\lambda}$ is the number of time slots within the active slots' coverage of the γ forwarding nodes after partition, λ is the number of segments in which the slot is not continuous after the division. The length of the j-th sleep slot is $t^s_{i,\gamma,\lambda,j}|j\epsilon 1..\lambda$ and it satisfies $\sum_{j=1}^{\lambda} t^s_{i,\gamma,\lambda,j} = N - t^a_{i,\gamma,\lambda}$. And μ_k represents the length of the k-th active slots after partition.

Proof: When $\gamma > 1$, the M consecutive active slots are randomly divided into λ segments. There are $t^a_{i,\gamma}$ slots with a delay of 1 before partition, and the delay of the sleep slot in the jth segment is $\sum_{k=2}^{t^s_{i,\gamma,j}+1} k$. After the division, there are $t^a_{i,\gamma,\lambda}$ slots with a delay of 1 and the delay of the sleep slot in the jth segment is $\sum_{k=2}^{t^s_{i,\gamma,\lambda,j}+1} k$. At this time, the calculation formula of $\Delta\bar{d}_i$ is shown as

$$\begin{aligned}\Delta\bar{d}_i &= \frac{\left(t^a_{i,\gamma} + \sum_{j=1}^{\beta}\sum_{k=2}^{t^s_{i,\gamma,j}+1} k\right)}{N} \\ &\quad - \frac{\left(t^a_{i,\gamma,\lambda} + \sum_{j=1}^{\lambda}\sum_{k=2}^{t^s_{i,\gamma,\lambda,j}+1} k\right)}{N} \\ &= \frac{\left(t^a_{i,\gamma} - t^a_{i,\gamma,\lambda} + \sum_{j=1}^{\beta}\sum_{k=2}^{\xi_j+1} k - \sum_{j=1}^{\lambda}\sum_{k=2}^{\mu_j+1} k\right)}{N}.\end{aligned}$$

In the best case after partition, the number of the active slots of the γ child nodes is $t^a_{i,\gamma,\lambda} = N$, which is distributed throughout the whole period. The delay at any slot is 1 and latency is 0. So the average delay to the next hop is 1. This situation requires $\gamma \times M \geq N$. At this time, the calculation formula of $\Delta\bar{d}_i$ is shown as

$$\Delta\bar{d}_i = \frac{\left(t^a_{i,\gamma} + \sum_{j=1}^{\beta}\sum_{k=2}^{\xi_j+1} k\right)}{N} - 1.$$

In the worst case, there is a significant change in active slots coverage for a single forwarding node, but the total $T^c_{\{1..j\}}$ remains unchanged. Due to $t^a_{i,\gamma} = t^a_{i,\gamma,\lambda}$, the average delay of node i doesn't decrease. At this point $\Delta\bar{d}_i$ is shown as

$$\Delta\bar{d}_i = 0.$$

□

$$\Delta D_i = \begin{cases} \dfrac{(2N-4M+1)(2N-2M+1-r)+(2M-1)(N-2M+1-r))r}{2N^2}, & \textit{generally case} \\ \dfrac{(2N-2M+1-r)r}{N}, & \textit{best case} \\ \dfrac{(2N-2M+1-r)r}{2N}, & \textit{worst case} \end{cases} \tag{19}$$

$$\begin{aligned}\Delta\mathcal{D}_i &= \frac{1}{N} \times \frac{(2N-2M+1-r)r}{2N} + \frac{N-2M}{N} \\ &\quad \times \left(\frac{(2N-2M+1-r)r}{N} \times \frac{1}{4} + \sum_{q=0}^{N-2M} \frac{2q^2 + 2(N-2M)(2r-|q-r|-q) - (2r-|q-r|)^2 + 2r - |q-r|}{2N} \right. \\ &\quad \left. \times \frac{1}{2} + \sum_{q=0}^{N-2M} \frac{2q^2 - 2(N-2M)q + (N-2M)^2 + N - 2M}{2N} \times \frac{1}{4} \right) + \frac{2M-1}{N} \times \frac{(2(N-2M+p) - r + 1)r}{2N} \\ &\approx \frac{(2N-2M+1-r)r}{2N^2} + \frac{(N-2M)(2N-2M+1-r)r}{N^2} + \frac{(2M-1)(N-2M+1-r)r}{2N^2} \\ &= \frac{(2N-4M+1)(2N-2M+1-r)+(2M-1)(N-2M+1-r))r}{2N^2}.\end{aligned}$$

Theorem 8: The continuous active slot is randomly divided into discontinuous active slots, and the calculation formula of $\Delta\mathcal{D}_i$ in general is as shown in Eq. (21):

Proof: The process of random partition is to first randomly select a slot s_i^d as a partition in the active slots. In front of slot s_i^d are $\left(s_i^d - s_i + N\right) mod N$ active slots from s_i to $s_i^d - 1$. Then set the active slot state after slot s_i^d to sleep. In all the sleep slots, select $M - \left(s_i^d - s_i + N\right) modN$ slots to be active. The total number of active slots remains constant as M. For example, if divided into two active slots separated by one sleep slot, it can only play a weak role in reducing latency. However, when it is divided into M segments active slots that are evenly distributed in the period, the reduction of latency is the largest, but the routing table length will increase by more. Therefore, in order to achieve the optimal latency and routing table, it is usually divided into two segments or three segments of active slots, and the distribution of active slots is as average as possible. The following is the calculation process of the reduced average delay when it is divided into two segments and three segments, respectively. First, select a slot in the active slots as s_i^d, and the probability of each slot being selected is equal to $\frac{1}{M}$. When $\left(s_i^d - s_i + N\right) modN = M - 1$ is satisfied, it is divided into two segments. Then select one slot from the other $N - M + 1$ sleep slots and set the state as active. The probability of being selected for each slot is equal to $\frac{1}{N-M+1}$. At this time, the calculation formula of $\Delta\mathcal{D}_i$ is as shown in Eq. (22):

$$\begin{aligned}\Delta\mathcal{D}_i &= \frac{1}{N}\left(\sum_{k=2}^{N-M+1} k - \frac{2\sum_{j=2}^{N-M+1}\sum_{k=2}^{j} k}{N-M+1}\right) \\ &= \frac{(N-M)(N-M+3)}{2N} - \frac{2\sum_{j=2}^{N-M+1}\sum_{k=2}^{j} k}{N(N-M+1)} \\ &= \frac{(N-M)(N-M+3)}{2N} - \frac{(N-M)(N-M+5)}{3N} \\ &= \frac{(N-M)(N-M-1)}{6N}\end{aligned} \tag{22}$$

When $\left(s_i^d - s_i + N\right) mod\ N = M - 2$, it indicates that it is divided into two or three segments active slots. Next, select two slots from $N - M + 2$ sleep slots and set the state as active. There are C_{N-M+2}^2 possible situations. At this time, the calculation formula of $\Delta\mathcal{D}_i$ is shown as Eq. (23), as shown at the bottom of the next page.

Based on Eq. (22) and Eq. (23), the two add the average to get $\Delta\mathcal{D}_i$ in general case can be inferred as shown in Eq. (21).

B. ROUTING TABLE STORAGE

Figure 24 shows the awaken sequence diagram of one parent node and three forwarding nodes.

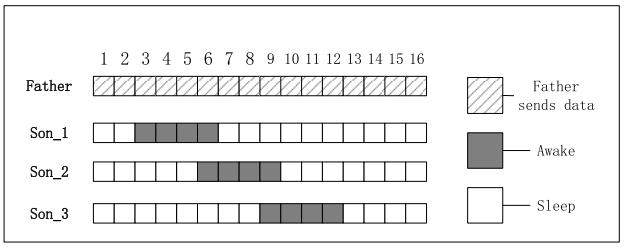


FIGURE 24. The example of definition 4.

As shown in Figure 24, the number of active slots is four. The starting active slot of node 1 is slot 3, the starting active slot of node 2 is slot 6, and the initial active slot of node 3 is slot 9. The start slot of the first segment of NDI is slot 1. At slot 1-5, the delay of sending packets from the parent node to the child node 1 is the shortest, so the forwarding node is the child node 1 when the data packet is sent in these slots. At slot 6, both child node 1 and child node 2 are in the active state, but slot 6 still belongs to the first segment of NDI, so the forwarding node is still child node 1. At slot 7-8, node 2 is always in the active state, so packets can be sent immediately without waiting, where latency is 0. Therefore, the forwarding node is child node 2. At slot 9, although both node 2 and node 3 are in the active state, the previous slot 8 belongs to the second segment NDI, so the forwarding node is the child node 2. Similarly, there is node 3 in the active state during slot 10-12, so the forwarding node is child node 3. At slot 13-16, the closest active slot is slot 3. At slot 3, node 1 is in the active state, so the forwarding node is the child node 1. Table 7 shows the routing table of the parent node. As can be seen from Table 7, the parent node has four NDIs, so there are four routing records, the corresponding start delay and end delay are calculated.

$$\Delta\bar{d}_i = \begin{cases} \dfrac{\left(t_{i,\gamma}^a - t_{i,\gamma,\lambda}^a + \sum_{j=1}^{\beta}\sum_{k=2}^{\xi_j+1} k - \sum_{j=1}^{\lambda}\sum_{k=2}^{\mu_j+1} k\right)}{N}, & generally \\ \dfrac{\left(t_{i,\gamma}^a + \sum_{j=1}^{\beta}\sum_{k=2}^{\xi_j+1} k\right)}{N} - 1, & best\ case \\ 0, & worst\ case \end{cases} \tag{20}$$

$$\Delta\mathcal{D}_i = \frac{2(N-M)^4 + 14(N-M)^3 + 25(N-M)^2 - 5(N-M)}{12(N-M+2)(N-M+1)N} \tag{21}$$

TABLE 7. The example of definition 4.

Start time slot	Start delay	End delay	Next hop
1	3	1	Son_1
7	1	1	Son_2
10	1	1	Son_3
13	7	4	Son_1

According to the routing table, the delay diagram at each time slot can be drawn as follows: the horizontal axis(X) represents the slot, the vertical axis(Y) shows delay. First, coordinates of several points are determined according to [*startT*, *startD*] in the table. Then, according to the delay non-incremental principle, the delay in each segment of NDI is always equal to one or one less than the delay of the previous slot, and both are continuous. This is the same thing as making a straight line with a slope of -1 from each of four points until the vertical delay of the point is equal to 1. A line parallel to the X-axis is made from this point after the delay is equal to one. According to Table 7, Figure 25 shows the delay diagram of the parent node in each time slot.

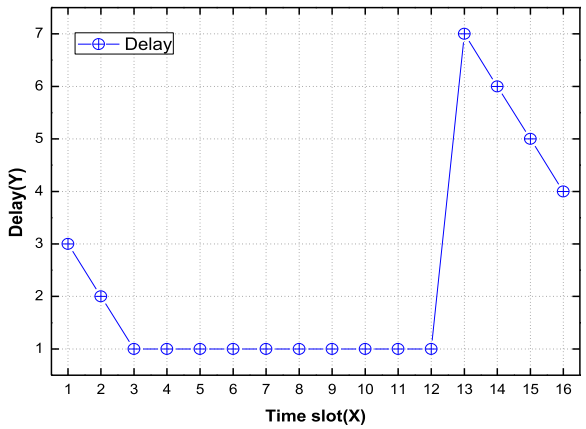


FIGURE 25. The delay of father node.

Theorem 9: There are N time slots and M consecutive active slots in a period. The initial active slots of the candidate forwarding nodes are distributed in s^a slots. Then sort these s^a slots from smallest to largest. The k-th slot is $s_k^a (k \in [1, s^a])$, and the slot of the active slots corresponding to s_k^a is e_k^a. So the number of routing tables (r) satisfies Eq. (24):

$$r \leq s^a + 1 \tag{24}$$

Proof: The first route starts with slot 1, the start delay is s_1^a, the end slot is slot e_1^a, the end delay is 1, and the next hop is the child node whose active slot is s_1^a. The start slot of the next route is the next slot at the end slot of the previous route, so the start slot of the second route is $e_1^a + 1$. If $e_1^a \leq s_2^a$, the start delay of the second route is $s_2^a - e_1^a$, the end slot is e_2^a, the end delay is 1, and the next hop is the child node whose active slot is s_2^a. Otherwise if $s_2^a < e_1^a < e_2^a$, the start delay of the second route is 1. Next, it is judged whether the start slot s_3^a after s_2^a satisfies the condition of $s_3^a < e_1^a$. If not, the end slot is e_2^a and the end delay is 1. If the condition is satisfied, continue to judge the next s_k^a until the condition is not satisfied. At this point, the end slot is e_{k-1}^a and the end delay is 1. The start slot of the third route is the slot $e_{k-1}^a + 1$, and the start delay is $\max\left\{s_k^a - e_{k-1}^a, 1\right\}$. By this method, the end slot is the last slot corresponds to the nearest active slot after slot $e_{k-1}^a + 1$, and the end delay is 1. If and only if the last end slot $e_{s^a}^a < N$, there might be the $s^a + 1$ route. The start slot of the $s^a + 1$ route is slot $e_{s^a}^a + 1$, and the start delay is $\max\left\{s_1^a - e_{s^a}^a + N + 1, 1\right\}$, the end slot is slot N, and the end delay is s_1^a. In summary, the number of routing tables r satisfies Eq. (24). □

Theorem 10: Under the premise of theorem 9, the number of valid nodes in the network is adjusted from v^{ori} to $v^{\text{ori}} - v^{re}$, and these v^{re} nodes are removed from the forwarding node set. The k-th node in these v^{re} nodes have v_k^{ch} child nodes ($k \in [1, v^{re}]$). If the first active slots of the child nodes are distributed in different s_k^{ch} slots, $1 \leq s_k^{ch} \leq v_k^{ch}$, then the reduction of the routing table length satisfies Eq. (25):

$$v^{re} \leq \Delta r \leq \left(v_k^{ch} + 2\right) \times v^{re} \tag{25}$$

Proof: The minimum reduction of the routing table length is when the reduced v^{re} nodes are all parent nodes. At this point, only the routing tables of these v^{re} nodes are involved, but the routing tables of its neighbor nodes are not involved. If the initial active slots of the v_k^{ch} child nodes are distributed in the same time slot, that is, $s_k^{ch} = 1$, it can be seen from Theorem 9 that the k-th node has at least one route. So, reducing v^{re} nodes can at least reduce v^{re} routes. It is considering that the maximum reduction of the routing table length is when the reduced v^{re} nodes are not only the

$$\begin{aligned}\Delta\mathcal{D}_i &= \frac{1}{N}\left(\sum_{k=2}^{N-M+1} k - \frac{\sum_{j=2}^{N-M+1}\left(3\sum_{k=2}^{j} k + 2\sum_{m=2}^{N-M+1}\sum_{n=2}^{m} n\right)}{C_{N-M+2}^{2}}\right) \\ &= \frac{(N-M)(N-M+3)}{2N} - \frac{6\sum_{j=2}^{N-M+1}\sum_{k=2}^{j} k + 4\sum_{j=2}^{N-M+1}\sum_{m=2}^{j}\sum_{n=2}^{m} n}{(N-M+2)(N-M+1)N} \\ &= \frac{\left(2(N-M)^2+7\right)(N-M+3)(N-M)}{6(N-M+2)(N-M+1)N}\end{aligned} \tag{23}$$

parent nodes but also the relay nodes of the neighbor nodes. In this case, in addition to the reduction of the routing table of these v^{re} nodes, it also reduces the number of routes for the parent nodes of the relay nodes. If the initial active slots of v_k^{ch} child nodes do not coincide, it is distributed in v_k^{ch} slots, that is, $s_k^{ch} = v_k^{ch}$. According to Theorem 9, there is no more than n_k+1 routes in the kth node, so reducing v^{re} nodes can reduce $\sum_{k=1}^{v^{re}} \left(v_k^{ch}+1\right)$ routes at most. For the parent nodes of the v^{re} nodes, the maximum number of v^{re} routes can be reduced, so the number of $\sum_{k=1}^{v^{re}} \left(v_k^{ch}+1\right) + v^{re} \leq \left(v_k^{ch}+2\right) \times v^{re}$ routes can be reduced by reducing v^{re} nodes at most. □

VI. ANALYSIS OF EXPERIMENTAL RESULTS

A. EXPERIMENT SETUP

In this section, we through two specific examples to illustrate the motivation of the Self-adjustable Active Sequence (SAC) scheme proposed in this paper.

B. EXPERIMENT ONE

This experiment is just the same as that in research motivation, so there are a lot of things that have appeared before, so we will not repeat them here, just add something that it did not cover in detail. The Figure 2 represents the network topology model, and Figure 3 shows the slot state of each node.

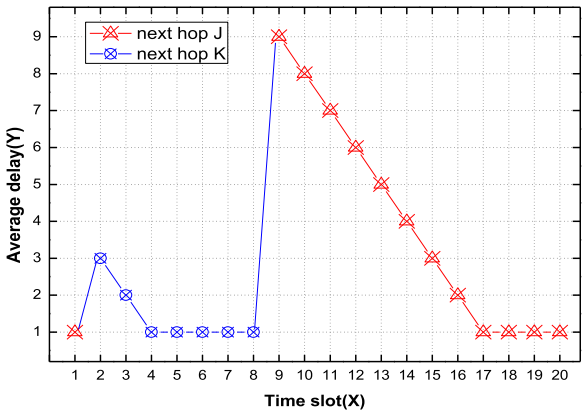


FIGURE 26. Delay of node A.

Next is a demonstration of the delay calculation method. node K and J are the forwarding relay nodes of node A. Node K is active when node A is ready for data at slot 1, then node A can send data at slot 1. Currently, the delay is 1 and the next hop is node K. If the data is ready at slot 2, but both node K and J are sleep. Node J is active until slot 4, and it means node A can only send data until slot 4. At this moment, the delay is 3 and the next hop is node J. Figure 26 shows the delay of node A in each time slot. Where the different color symbols represent different relay node: red represents node J, and blue means node K. There is a large delay gap among different slots, with the maximum reaches 9 and the minimum reaches 1. Calculate the average delay is 2.95, which means that it takes an average of 2.95 slots for the next hop to successfully receive the data, and an average of 1.95 slots to waiting before sending.

Not only the delay of different slots in the same node varies greatly, but also the average delay of different nodes. The maximum average delay is 7 and the minimum is 1. The average delay of all nodes is 3.75, which means that each node sends packets at any slot and requires an average of 3.75 slots before the next hop to receive them successfully, with an average latency of 2.75 slots.

The first scheme is to divide continuous active slots into multi-segment discontinuous active slots. Based on Figure 3, Figure 5 shows the slot state of each node after partition. Overall, the average delay is 2.324, and average latency is 1.324. Compared with the situation of the continuous active slots, 1.426 slots are reduced, which reduced the latency by 51.9%.

Figure 27 shows the histogram of continuous and discontinuous average delay. The gray shaded bar represents the average delay in the case of discontinuous active slots, and the red diagonal bar means the continuous active slots. We can draw the conclusion that the improvement effect of division is noticeably since the delay reduction effect of most nodes is obvious.

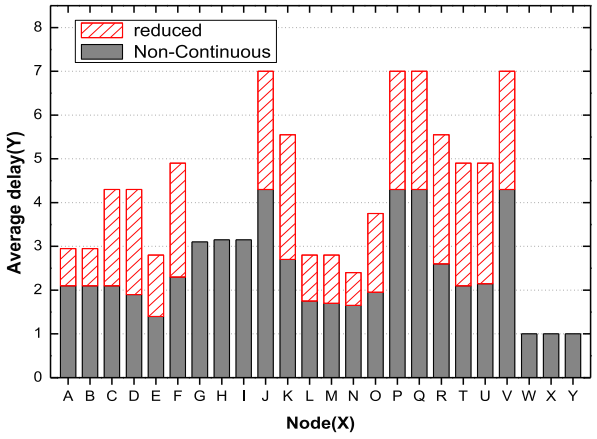


FIGURE 27. Average delay comparison histogram.

The second scheme is to add the active slots in the far sink region. The average delay at $M = 5$ has been obtained previously, and the experiment of $M = 8$ and $M = 10$ will be performed here. The node state diagrams when $M = 8$ and when $M = 10$ are shown as Figure 7(a) and Figure 7(b).

Figure 28 shows the comparison diagram of average delay under different M values. With the increase of M, the average delay becomes smaller and smaller. We can notice that the three nodes W, X and Y can send data directly without waiting, which means latency is 0 and delay is 1. The delay has been minimized and cannot be reduced any more. Except the three nodes, the average delay of other nodes has been reduced to different degrees. When M is increased from 5 to 8, the delay can be reduced by nearly 60% at most, 12% at least, and 37% on average. Then increase M from 5 to 10 can

reduce the delay by 70% at most, 24% at least, and 52% on average. By comparison, when increasing M from 8 to 10, the delay can be reduced by 37% at most, 1% at least, and 23% on average. Moreover, when M increases from 5 to 8, the decreases at least twice as much as when M increases from 8 to 10. We can reflect that when the value of M is large enough, adding active slots has a less significant effect on delay reduction.

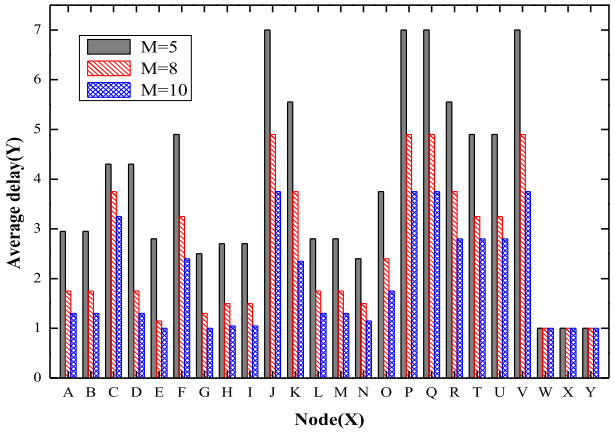


FIGURE 28. Average delay comparison with different M.

The next experiment is further reducing the number of candidate nodes in the remote sink area. As shown in Figure 11, in the outermost layer and the second outer layer respectively reduced two candidate nodes, and the reduced four nodes are A, I, J, and P. Table 8 shows the comparison of routing tables of node B and E before and after reducing candidate nodes. After removing node J from the candidate nodes set, node B can only send data to node K, resulting in only one record in the routing table.

TABLE 8. Contrast about reducing nodes.

Node	Before		After	
	Starting slot	Next hop	Starting slot	Next hop
B	1	K	1	K
	7	J		
	14	K		
H	1	O	1	O
	8	P		
	17	O		

The next step is to consider the energy residual in the network. The strategy is to make $M = 10$ of the outermost and the second outer nodes, Figure 9 represents the active slot state after increasement. The average delay of the outermost nodes has been significantly reduced, with the largest decrease by nearly 70% and the smallest by 24%, and with an average reduction of 56%. Figure 29 shows the average delay comparison before and after reduction, where the red shaded portion represents the reduction. The average delay for all nodes is 3.092, which decreased by 0.658 slots, or 18%, compared to before it was added.

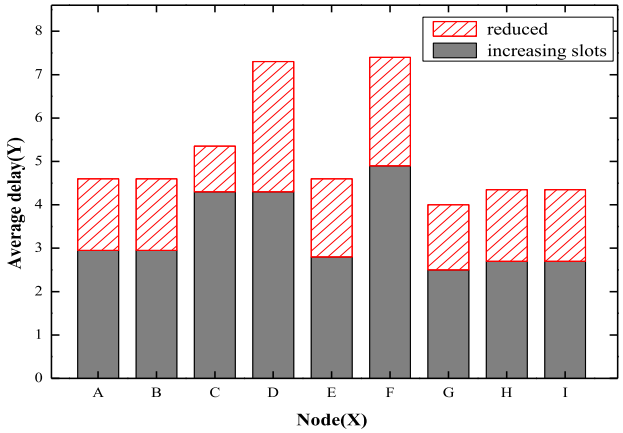


FIGURE 29. Average delay Comparison after increasement.

C. EXPERIMENT TWO

In this experiment, the relevant parameters are set as $K = 53$, $M = 5$, $N = 20$, $\frac{M}{N} = 25\%$. Node S is the sink node too. The network topology model is shown in Figure 30. Figure 31 shows the time slot state, where the first slot is selected randomly.

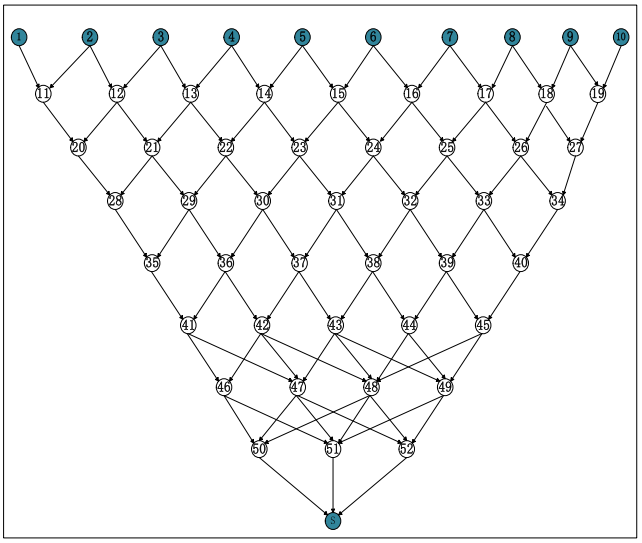


FIGURE 30. Network topology.

We will take node 2 as an example, which has the forwarding nodes as node 11 and 12. When there are packets to be sent at slots 11-14, next hop is node 12, which has the delay of 1. At other slots, next hop is node 11. The delay at slot 6-14 is 1, and the average delay is 4.3.

Figure 32 represents the polyline graph of the average delay of each node. The maximum average delay is 7, and there are 12 nodes with only one forwarding node. The aver-

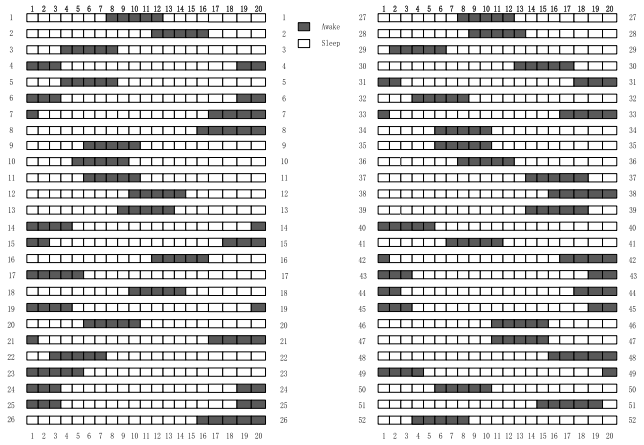


FIGURE 31. Duty cycle time slot.

age delay of the three nodes closest to the sink node is the minimum. The average delay of all nodes is 4.34.

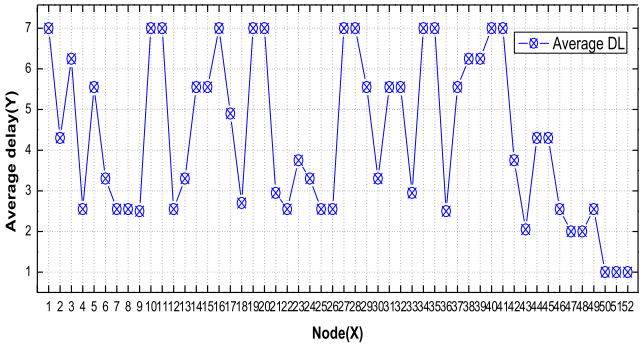


FIGURE 32. Average delay.

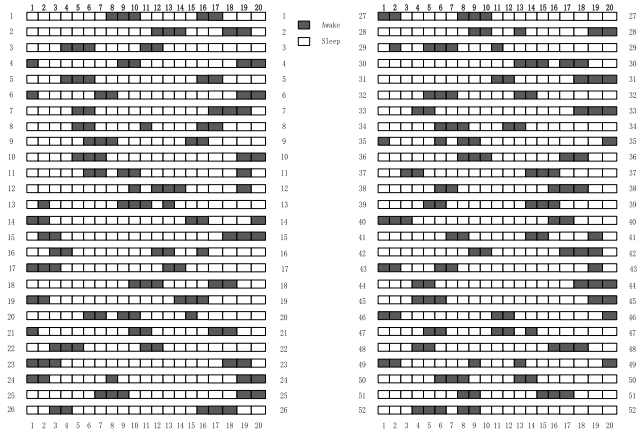


FIGURE 33. Duty cycle time slot with interval.

Next, the continuous active time slots are divided into discontinuous active time slots. Figure 33 shows the active time slots after the division. The average delay of all nodes is 2.89, which is 1.45 slots less than the continuous active slots and reduced by 33%.

Figure 34 shows the histogram comparison of the two. The gray shaded bar represents the average delay after improvement, and the red diagonal bar represents the reduced delay compared with the original delay. If delay increases after improvement, slash bars are not displayed. The delay of most nodes is significantly reduced, and the average delay is generally reduced.

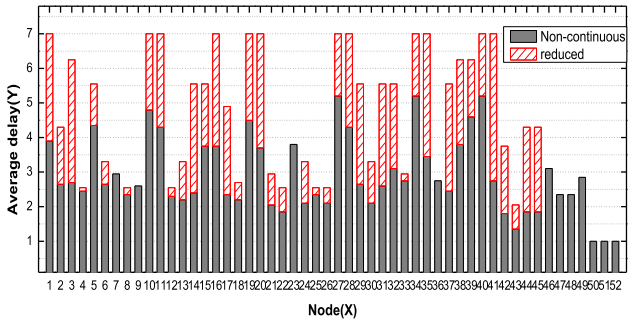


FIGURE 34. Average delay comparison histogram.

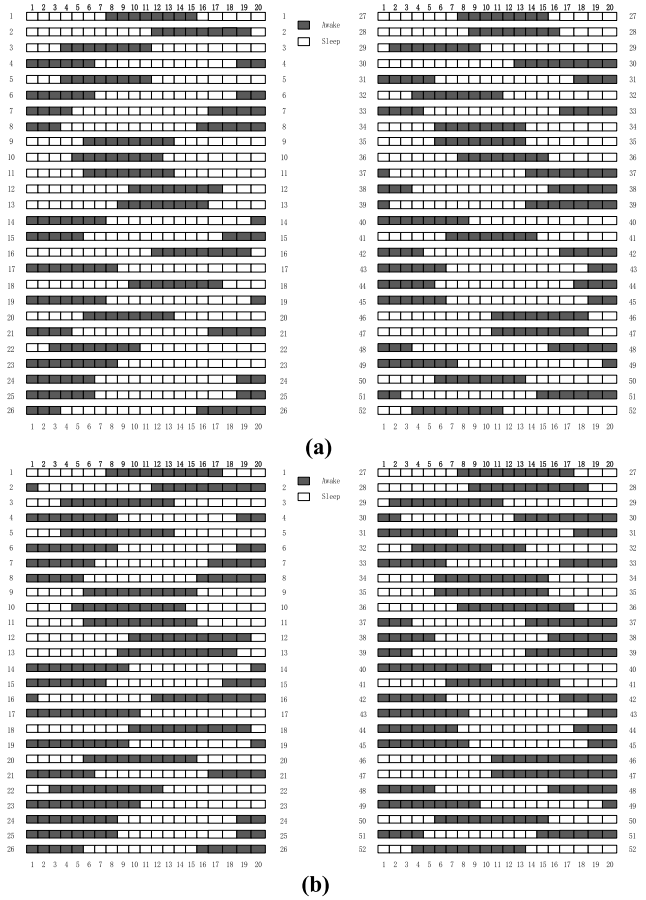


FIGURE 35. Duty cycle time slot: (a) $M = 8$, (b) $M = 10$.

The next step is to increase M by adding several time slots at the end of the original active slots. The previous experiments have calculated the average delay when $M = 5$, and

the average delay when $M = 8$ and $M = 10$ are calculated in this step. Figure 35 (a) and Figure 35 (b) represent the time slot state of each node when $M = 8$ and $M = 10$ respectively.

Figure 36 shows the histogram of the average delay between $M = 8$ and $M = 5$. Where the gray shaded bar represents the average delay of $M = 8$, the red slash bar represents the reduced delay, and the red slash bar plus the gray shaded bar is the delay of $M = 5$. When $M = 8$, the average delay after adding active slots is 2.87, which is 1.47 slots less than the average delay of $M = 5$, and which is also reduced by 33.87%. Furthermore, the average delay is 2.19 when = 10. Compared with the delay at = 5, it is reduced by 2.15 slots, which is reduced by 50%. Compared with the delay of = 8, it is reduced by 0.68 slots, which is reduced by 24%.

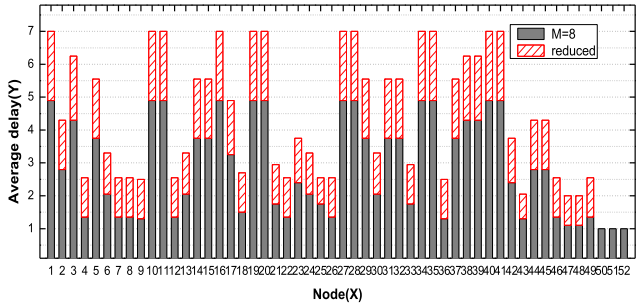


FIGURE 36. Average delay comparison between $M = 8$ and $M = 5$.

Figure 37 and Figure 38 respectively show the polyline graph and histogram of average delay under different M values. The black polyline representing $M = 5$ is always above the other two lines, and the blue polyline indicating $M = 8$ is still in the middle of the other two lines. As can be seen from Figure 38, when the M value increases from 5 to 8 and then to 10, there are two large drops in the delay. The more active slots are added, the more delay is reduced. In a word, the average delay is 2.19 when $M = 10$. Compared with the delay at $M = 5$, it is reduced by 2.15 slots, which is reduced by 50%. Compared with the delay of $M = 8$, it is reduced by 0.68 slots, which is reduced by 24%.

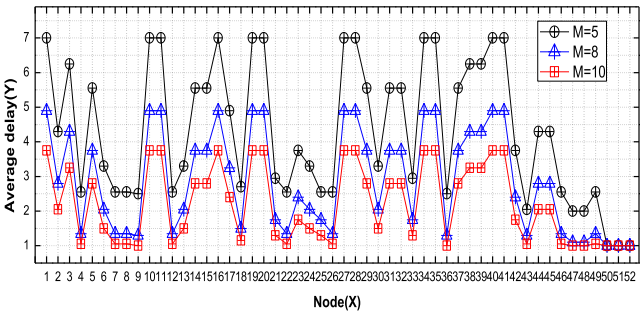


FIGURE 37. Average delay line comparison with different M.

The next experiment reduces forwarding relay nodes in the far sink region based on the $M = 10$ in Figure 35

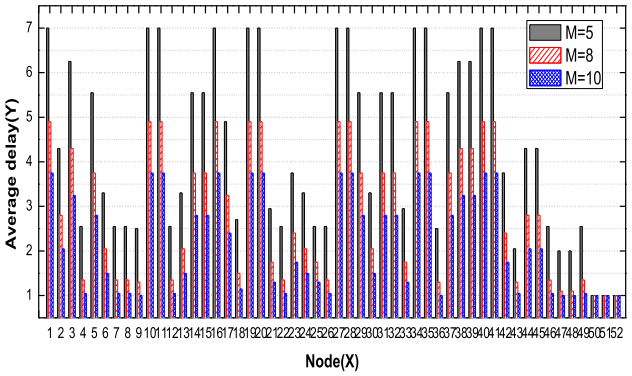


FIGURE 38. Average delay comparison with different M.

(b). Figure 39 shows the network topology after reduction. The dotted line between the node and the neighbor nodes indicating the node subtracted randomly. As can be seen from the figure, at this time $K = 45$ and these removed eight nodes are numbered 2, 9, 12, 18, 21, 26, 29 and 33, respectively.

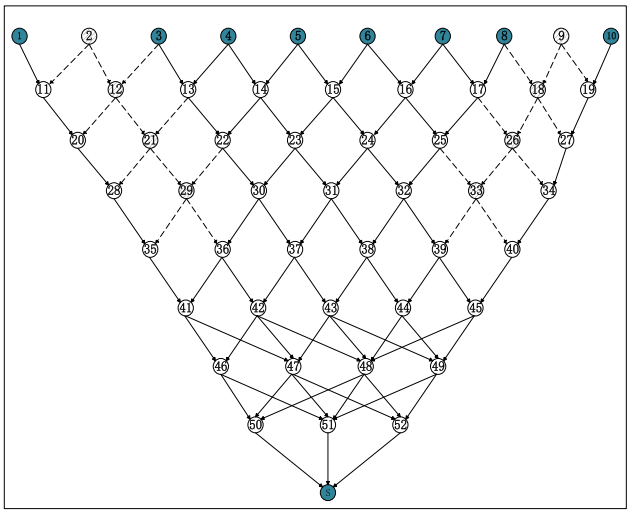


FIGURE 39. Network topology after reducing the forwarding relay nodes.

Table 9 presents the changes in the routing table before and after the reduction. As can be seen from Table 9, node 2 originally has three routing records. After node 12 is made unavailable in the network, the neighbor nodes 2, 3, 20, and 21 are notified. The neighbor nodes adjust the routing table. After the adjustment, there is only one routing table record for node 3, which is reduced by two. The routing tables of both node 2 and 12 are all emptied. After reducing the eight nodes in the remote sink area, there are 14 nodes that affect the routing table. There are 40 routing table changes in these nodes, among which there are only six routing records after improvement, and 34 records are reduced.

In order to solve the energy hole problem, since the duty cycle diagram given in Figure 31, adding active nodes in the outermost layer and the second outer layer nodes. The outermost layer and the second outer layer nodes are numbered

TABLE 9. Routing table comparison after reducing nodes.

Node	Before		After	
	Starting slot	Next hop	Starting slot	Next hop
2	1	11	Null	
	16	12		
	20	11		
3	1	13	1	13
	19	12		
	20	13		
8	1	17	1	17
	11	18		
	20	17		
9	1	19	Null	
	10	18		
	20	19		
12	1	21	Null	
	7	20		
	16	21		
13	1	21	1	22
	7	22		
	13	1		
17	1	25	1	25
	9	26		
18	1	26	Null	
	6	27		
	18	26		
21	1	29	Null	
	12	28		
	19	29		
22	1	30	1	30
	3	29		
	12	30		
25	1	33	1	32
	7	32		
	14	33		
26	1	33	Null	
	7	34		
	16	33		
29	1	35	Null	
	16	36		
	18	35		
33	1	40	Null	
	10	39		

from 1 to 19. Here, the number of active slots is increased to $M = 8$ and $M = 10$ respectively, and the other nodes remain unchanged $M = 5$. Figure 40 shows the time slot state diagram of the node after the active slots are added. In Figure 40(a), increase M to 8, and in Figure 40(b), increase M to 10, whereas the working slots of other nodes remain the same.

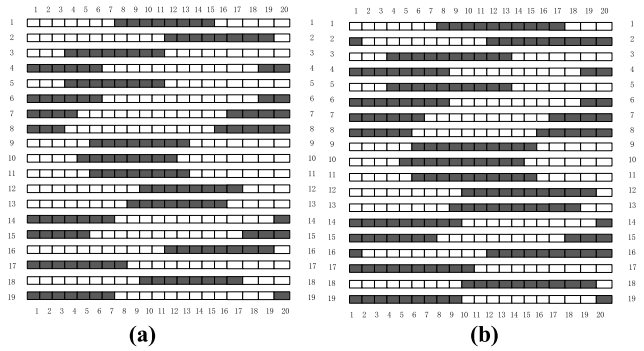


FIGURE 40. Network topology duty cycle time slot after increasing the active time slot (a) $M = 8$, (b) $M = 10$.

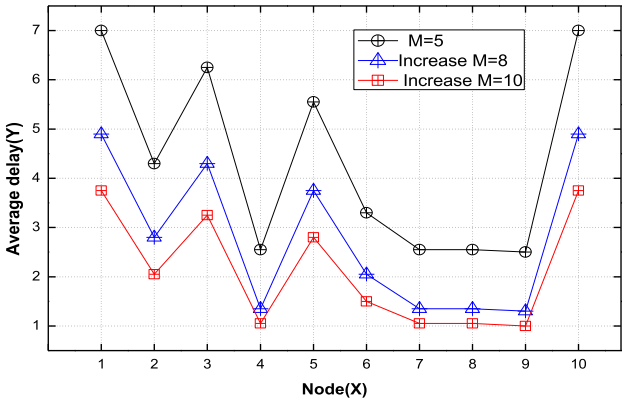


FIGURE 41. Average delay line comparison after increasing the active time slot.

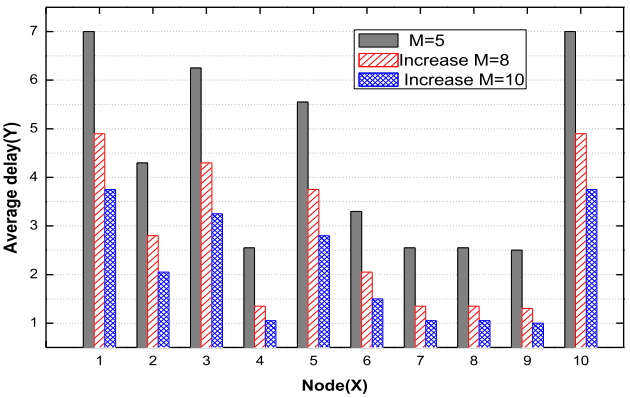


FIGURE 42. Average delay comparison after increasing the active time slot.

Figure 41 and Figure 42 respectively show the polyline comparison and the histogram comparison of average delay in three cases. It is evident that the average delay of ten nodes numbered from 1 to 10 is reduced. As can be seen from the chart, when $M = 8$, the average delay of all nodes is 4.04, which is 0.3 slots less than before the improvement, which

is reduced by 6.9%. And when $M = 10$, the average delay of all nodes is 3.91, which is 0.43 slots less than before the improvement, which is reduced by 9.9%. By using the energy hole problem to increase the awake time slot of the remote sink area nodes can reduce the delay.

VII. CONCLUSION

The above formula derivation and experiments prove that Self-adjustable Active Sequence (SAC) scheme has certain effects in lower latency, increasing network lifetime and reducing routing table length. The optimization algorithms proposed by SAC scheme are tailored for several cases with different delays. When the delay in the network is large, the main goal currently is to reduce the delay. At this point, the SAC scheme divides the continuous active slots into multi-segments slots with smaller length. It sacrifices a small portion of network lifetime performance and route lookup time in exchange for greater delay reduction. In general, if and only if the continuous active slot is divided into several segments, the delay can be reduced by 33%. In another case, when the main goal is not to reduce the delay but to improve the network energy utilization, the method of using the residual energy to increase the number of active slots can be adopted. We notice that there is a surplus of energy in the far sink region, consequently making full use of the residual energy does not reduce the network lifetime. Some segments of active slots can be connected into continuous active slots so that achieve the purpose of reducing latency as well as reducing the routing table length. Under the condition of experiment two, it can further reduce the delay by 29%. Moreover, when the delay is already small enough, the main goal is how to improve the routing speed. Available data suggest the potential to reduce the size of the forwarding nodes set can effectively increase the routing speed. In experiment one and experiment two, a scheme of reducing several nodes randomly was adopted to reduce the length of routing table. The latter's routing table length is reduced by 29%.

Although the SAC scheme proposed in this paper had done a lot of work to get the above conclusions, the SAC scheme still has a lot of work to do in the future. In the author's opinion, the comprehensive expression that represent the relationship between latency, routing table length and network lifetime remains to be elucidated in more detail. Then, it is a big challenge that how to adjust active slots more automatically. Imagine a scenario, input parameters in the network, such as node distribution location, network topology, initial active sequence and so on, then can automatically adjust delay, routing efficiency and network life to optimal state by SAC scheme. This will be the focus of follow-up work. What's more, there are other more directions to research. The performance metric used to measure delay in the paper is the average delay, which is the average of the delays that all nodes have reached the sink node after multiple hops. There is a high probability to minimum the maximize delay as a new metric for delay. To calculate the theoretical minimum of the delay of the node with the maximize delay among all nodes, which can guarantee fairness. Another development prospect is the introduction of differentiated network services. First, prioritize the different data and set a priority threshold. The higher the priority, the more urgent the data. If the priority of the forwarded data is higher than this threshold, it can actively wake up the forwarding node instead of waiting for the arrival of the active slots of the forwarding node. This method can guarantee the transmission time of vital data. In conclusion, we should build on existing experiments and data to refine the proposed SAC scheme.

REFERENCES

[1] A. Escobar-Molero, "Improving reliability and latency of wireless sensor networks using concurrent transmissions," *At-Automatisierungstechnik*, vol. 67, no. 1, pp. 42–50, Jan. 2019.

[2] Y. Liu, M. Ma, X. Liu, N. Xiong, A. Liu, and Y. Zhu, "Design and analysis of probing route to defense sink-hole attacks for Internet of Things security," *IEEE Trans. Netw. Sci. Eng.*, to be published, doi: 10.1109/TNSE.2018.2881152.

[3] X. Deng, J. Luo, L. He, Q. Liu, X. Li, and L. Cai, "Cooperative channel allocation and scheduling in multi-interface wireless mesh networks," *Peer-Peer Netw. Appl.*, vol. 12, no. 1, pp. 1–12, Jan. 2019.

[4] N. Nguyen, B. Liu, V. Pham, and T. Liou, "An efficient minimum-latency collision-free scheduling algorithm for data aggregation in wireless sensor networks," *IEEE Syst. J.*, vol. 12, no. 3, pp. 2214–2225, Sep. 2018.

[5] H. M. Abdulsalam, B. A. Ali, and E. AlRoumi, "Usage of mobile elements in Internet of Things environment for data aggregation in wireless sensor networks," *Comput. Elect. Eng.*, vol. 72, pp. 789–807, Nov. 2018.

[6] M. Huang, W. Liu, T. Wang, A. Liu, and S. Zhang, "A cloud-MEC collaborative task offloading scheme with service orchestration," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2019.2952767.

[7] Q. Liu, G. Wang, X. Liu, T. Peng, and J. Wu, "Achieving reliable and secure services in cloud computing environments," *Comput., Elect. Eng.*, vol. 59, pp. 153–164, Apr. 2017.

[8] Y. Liu, A. Liu, X. Liu, and M. Ma, "A trust-based active detection for cyber-physical security in industrial environments," *IEEE Trans. Ind. Informat.*, vol. 15, no. 12, pp. 6593–6603, Dec. 2019, doi: 10.1109/TII.2019.2931394.

[9] C. Zhang, R. Chen, L. Zhu, A. Liu, Y. Lin, and F. Huang, "Hierarchical information quadtree: Efficient spatial temporal image search for multimedia stream," *Multimed. Tools Appl.*, vol. 78, no. 21, pp. 30561–30583.

[10] H. Xiong, Y. Zhao, L. Peng, H. Zhang, and K.-H. Yeh, "Partially policy-hidden attribute-based broadcast encryption with secure delegation in edge computing," *Future Gener. Comput. Syst.*, vol. 97, pp. 453–461, Aug. 2019.

[11] S. Cai, Y. Zhu, T. Wang, G. Xu, A. Liu, and X. Liu, "Data collection in underwater sensor networks based on mobile edge computing," *IEEE Access*, vol. 7, pp. 65357–65367, 2019.

[12] F. Wang, W. Liu, T. Wang, M. Zhao, M. Xie, H. Song, X. Li, and A. Liu, "To reduce delay, energy consumption and collision through optimization duty-cycle and size of forwarding node set in WSNs," *IEEE Access*, vol. 7, pp. 55983–56015, 2019.

[13] Q. Wang, W. Liu, T. Wang, M. Zhao, X. Li, M. Xie, M. Ma, G. Zhang, and A. Liu, "Reducing delay and maximizing lifetime for wireless sensor networks with dynamic traffic patterns," *IEEE Access*, vol. 7, pp. 70212–70236, 2019.

[14] Q. Liu, Y. Guo, J. Wu, and G. Wang, "Effective query grouping strategy in clouds," *J. Comput. Sci. Technol.*, vol. 32, no. 6, pp. 1231–1249, Nov. 2017.

[15] Y. Liu, A. Liu, X. Liu, and X. Huang, "A statistical approach to participant selection in location-based social networks for offline event marketing," *Inf. Sci.*, vol. 480, pp. 90–108, Apr. 2019.

[16] J. Tan, W. Liu, T. Wang, S. Zhang, A. Liu, M. Xie, M. Ma, and M. Zhao, "An efficient information maximization based adaptive congestion control scheme in wireless sensor network," *IEEE Access*, vol. 7, pp. 64878–64896, 2019.

[17] T. Shu, W. Liu, T. Wang, Q. Deng, M. Zhao, N. N. Xiong, X. Li, and A. Liu, "Broadcast based code dissemination scheme for duty cycle based wireless sensor networks," *IEEE Access*, vol. 7, pp. 105258–105286, 2019.

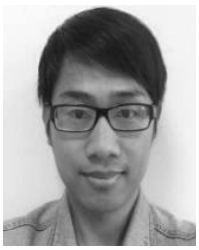
[18] X. Peng, J. Ren, L. She, D. Zhang, J. Li, and Y. Zhang, "BOAT: A block-streaming app execution scheme for lightweight IoT devices," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1816–1829, Jun. 2018.
[19] L. Yin, J. Gui, and Z. Zeng, "Improving energy efficiency of multimedia content dissemination by adaptive clustering and D2D multicast," *Mobile Inf. Syst.*, vol. 2019, Mar. 2019.
[20] X. Liu, A. Liu, T. Wang, K. Ota, M. Dong, Y. Liu, and Z. Cai, "Adaptive data and verified message disjoint security routing for gathering big data in energy harvesting networks," *J. Parallel Distrib. Comput.*, vol. 135, pp. 140–155, Jan. 2020.
[21] T. Wang, D. Zhao, S. Cai, W. Jia, and A. Liu, "Bidirectional prediction based underwater data collection protocol for end-edge-cloud orchestrated system," *IEEE Trans. Ind. Informat.*, to be published, doi: 10.1109/TII.2019.2940745.
[22] L. Hu, A. Liu, and M. Xie, "UAVs joint vehicles as data mules for fast codes dissemination for edge networking in smart city," *Peer-Peer Netw. Appl.*, vol. 12, no. 6, pp. 1550–1574, Nov. 2019.
[23] Y. Liu, X. Liu, A. Liu, N. Xiong, and F. Liu, "A trust computing based security routing scheme for cyber physical systems," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 6, 2019, Art. no. 61, doi: 10.1145/3321694.
[24] Q. Li, A. Liu, and T. Wang, "Pipeline slot based fast rerouting scheme for delay optimization in duty cycle based M2M communications," *Peer-Peer Netw. Appl.*, vol. 12, no. 6, pp. 1673–1704, Nov. 2019.
[25] X. Xiang, W. Liu, A. Liu, N. Xiong, Z. Zeng, and Z. Cai, "Adaptive duty cycle control based opportunistic routing scheme to reduce delay in cyber physical systems," *Int. J. Distrib. Sensor Netw.*, vol. 15, no. 4, Apr. 2019.
[26] Y. Liu, A. Liu, T. Wang, X. Liu, and N. N. Xiong, "An intelligent incentive mechanism for coverage of data collection in cognitive Internet of Things," *Future Gener. Comput. Syst.*, vol. 100, pp. 701–714, Nov. 2019.
[27] T. Wang, H. Luo, W. Jia, A. Liu, and M. Xie, "MTES: An intelligent trust evaluation scheme in sensor-cloud enabled industrial Internet of Things," *IEEE Trans. Ind. Informat.*, to be published, doi: 10.1109/TII.2019.2930286.
[28] O. Pandey and R. Hegde, "Low-latency and energy-balanced data transmission over cognitive small world WSN," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 7719–7733, Aug. 2018.
[29] G. Zhang, T. Wang, G. Wang, A. Liu, and W. Jia, "Detection of hidden data attacks combined fog computing and trust evaluation method in sensor-cloud system," *Concurrency Comput. Pract. Exper.*, Dec. 2018, doi: 10.1002/cpe.5109.
[30] X. Xiang, W. Liu, T. Wang, M. Xie, X. Li, H. Song, A. Liu, and G. Zhang, "Delay and energy efficient data collection scheme based matrix filling theory for dynamic traffic IoT," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, p. 168, Jun. 2019.
[31] M. Huang, W. Liu, T. Wang, Q. Deng, A. Liu, M. Xie, M. Ma, and G. Zhang, "A game-based economic model for price decision making in cyber-physical-social systems," *IEEE Access*, vol. 7, pp. 111559–111576, 2019.
[32] J. Ren, Y. Zhang, N. Zhang, D. Zhang, and X. Shen, "Dynamic channel access to improve energy efficiency in cognitive radio sensor networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 5, pp. 3143–3156, May 2016.
[33] W. Osamy, A. M. Khedr, A. Aziz, and A. A. El-Sawy, "Cluster-tree routing based entropy scheme for data gathering in wireless sensor networks," *IEEE Access*, vol. 6, pp. 77372–77387, 2018.
[34] X. Li, S. Liu, F. Wu, S. Kumari, and J. Rodrigues, "Privacy preserving data aggregation scheme for mobile edge computing assisted IoT applications," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4755–4763, Jun. 2019.
[35] Y. Yuan, W. Liu, T. Wang, Q. Deng, A. Liu, and H. Song, "Compressive sensing based clustering joint annular routing data gathering scheme for wireless sensor networks," *IEEE Access*, vol. 7, no. 1, pp. 114639–114658, 2019.
[36] J. Tan, W. Liu, M. Xie, H. Song, A. Liu, M. Zhao, and G. Zhang, "A low redundancy data collection scheme to maximize lifetime using matrix completion technique," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, no. 1, p. 5, 2019.
[37] W. Shi, W. Liu, T. Wang, Z. Zeng, and G. Zhi, "Adding duty cycle only in connected dominating sets for energy efficient and fast data collection," *IEEE Access*, vol. 7, no. 1, pp. 120475–120499, 2019.
[38] M. Shahzad, D. Nguyen, V. Zalyubovskiy, and H. Choo, "LNDIR: A lightweight non-increasing delivery-latency interval-based routing for duty-cycled sensor networks," *Int. J. Distrib. Sensor Netw.*, vol. 14, no. 4, Apr. 2018.
[39] T. Wang, H. Ke, X. Zheng, K. Wang, A. Sangaiah, and A. Liu, "Big data cleaning based on mobile edge computing in industrial sensor-cloud," *IEEE Trans. Ind. Informat.*, to be published, doi: 10.1109/TII.2019.2938861.
[40] Y. Chen, W. Liu, T. Wang, Q. Deng, A. Liu, and H. Song, "An adaptive retransmit mechanism for delay differentiated services in industrial WSNs," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, Nov. 2019, Art. no. 258, doi: 10.1186/s13638-019-1566-2.
[41] Y. Liu, A. Liu, and Z. Chen, "Analysis and improvement of send-and-wait automatic repeat-request protocols for wireless sensor networks," *Wireless Pers. Commun.*, vol. 81, no. 3, pp. 923–959, Apr. 2015.
[42] R. Kumar, R. Crepaldi, H. Rowaihy, A. Harris, III, G. Cao, M. Zorzi, and T. L. Porta, "Mitigating performance degradation in congested sensor networks," *IEEE Trans. Mobile Comput.*, vol. 7, no. 6, pp. 682–697, Jun. 2018.
[43] X. Sun, W. Liu, T. Wang, Q. Deng, A. Liu, N. Xiong, and S. Zhang, "Two-hop neighborhood information joint double broadcast radius for effective code dissemination in WSNs," *IEEE Access*, vol. 7, pp. 88547–88569, 2019.
[44] Y. Liu, Z. Zeng, X. Liu, X. Zhu, and M. Z. A. Bhuiyan, "A novel load balancing and low response delay framework for edge-cloud network based on SDN," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2019.2951857.
[45] Y. Wu, H. Huang, A. Liu, and T. Wang, "A risk defense method based on microscopic state prediction with partial information observations in social networks," *J. Parallel Distrib. Comput.*, vol. 131, pp. 189–199, Sep. 2019.
[46] A. Liu, Z. Zheng, C. Zhang, Z. Chen, and X. Shen, "Secure and energy-efficient disjoint multipath routing for WSNs," *IEEE Trans. Veh. Technol.*, vol. 61, no. 7, pp. 3255–3265, Sep. 2012.
[47] X. Liu, M. Zhao, A. Liu, and K. K. L. Wong, "Adjusting forwarder nodes and duty cycle using packet aggregation routing for body sensor networks," *Inf. Fusion*, vol. 53, pp. 183–195, Jan. 2020.
[48] L. Jiang, A. Liu, Y. Hu, and Z. Chen, "Lifetime maximization through dynamic ring-based routing scheme for correlated data collecting in WSNs," *Comput. Elect. Eng.*, vol. 41, pp. 191–215, Jan. 2015.
[49] Y. Liu, A. Liu, Y. Hu, Z. Li, Y. Choi, H. Sekiya, and J. Li, "FFSC: An energy efficiency communications approach for delay minimizing in Internet of Things," *IEEE Access*, vol. 4, pp. 3775–3793, 2016.
[50] Z. Chen, A. Liu, Z. Li, Y.-J. Choi, and J. Li, "Distributed duty cycle control for delay improvement in wireless sensor networks," *Peer-Peer Netw. Appl.*, vol. 10, no. 3, pp. 559–578, May 2017.
[51] Y. Hu, M. Dong, K. Ota, A. Liu, and M. Guo, "Mobile target detection in wireless sensor networks with adjustable sensing frequency," *IEEE Syst. J.*, vol. 10, no. 3, pp. 1160–1171, Sep. 2016.
[52] C. Zhu, V. C. M. Leung, J. J. P. C. Rodrigues, L. Shu, L. Wang, and H. Zhou, "Social sensor cloud: Framework, greenness, issues, and outlook," *IEEE Netw.*, vol. 32, no. 5, pp. 100–105, Sep. 2018.
[53] T. Ouyang, Z. Zhou, and X. Chen, "Follow me at the edge: Mobility-aware dynamic service placement for mobile edge computing," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 10, pp. 2333–2345, Oct. 2018.
[54] H. Teng, W. Liu, T. Wang, X. Kui, S. Zhang, and N. Xiong, "A collaborative code dissemination schemes through two-way vehicle to everything (V2X) communications for urban computing," *IEEE Access*, vol. 7, pp. 145546–145566, 2019, doi: 10.1109/ACCESS.2019.2940639.
[55] W. Liu, P. Zhuang, H. Liang, J. Peng, and Z. Huang, "Distributed economic dispatch in microgrids based on cooperative reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2192–2203, Jun. 2018.
[56] X. Chen, Q. Shi, L. Yang, and J. Xu, "ThriftyEdge: Resource-efficient edge computing for intelligent IoT applications," *IEEE Netw.*, vol. 32, no. 1, pp. 61–65, Feb. 2018.

XUAN LONG is currently pursuing the master's degree with the School of Computer Science and Engineering, Central South University, China. Her research interests include data process, crowd sensing networks, and wireless sensor networks.

WEI LIU received the Ph.D. degree in computer application technology from Central South University, China, in 2014. He is currently an Associate Professor and a Senior Engineer with the School of Informatics, Hunan University of Chinese Medicine, China. His research interests include software engineering, data mining, and medical informatics. He has published over 20 articles in the related fields.

TIAN WANG received the B.Sc. and M.Sc. degrees in computer science from Central South University, in 2004 and 2007, respectively, and the Ph.D. degree from the City University of Hong Kong, in 2011. He is currently an Associate Professor with the National Huaqiao University of China. His research interests include wireless sensor networks, social networks, and mobile computing.

ZHIWEN ZENG is currently a Professor with the School of Computer Science and Engineering, Central South University, China. His major research interests include wireless sensor networks and distributed computing.



MING MA is currently pursuing the Ph.D. degree with the Department of Computer Science, Stony Brook University, USA. His research interests include geometric modeling, digital geometric processing, and wireless sensor networks.