Mitigate the Bottleneck of Underwater Acoustic Sensor Networks via Priority Scheduling

Junjie Xiong Michael R. Lyu Kam-Wing Ng
Dept. of Computer Science & Engineering, The Chinese Univ. of Hong Kong, Shatin, Hong Kong

Abstract

Underwater acoustic sensor networks (UWASNs) are composed of underwater sensors that use sound to transmit information collected in the ocean. Since the sound speed is lower than radio wave, UWASNs suffer frommuch lower throughput and higher delay compared with terrestrial wireless sensor networks (TWSNs). Current methods manage to alleviate the bottleneck by replacing mutual handshakes with reservation mechanisms that consume lower overhead. However, their throughput improvement and delay reduction are very limited (e.g., the throughput is only 30% of the theoretical maximum for TLohi in 8-node networks), and most of their analysis and simulations are based on single-hop communication. In this work, we tackle the above challenges by proposing a priority scheduling approach for multi-hop topologies. First, we find that the scheduling problem of UWASNs is very different from that of TWSNs, and analyze the shortest schedule for the whole UWASN. Then, we design an efficient priority scheduling protocol at the MAC layer. Our approach performs parallel transmissions and prioritizes the scheduling by allocating longer time to heavier-traffic nodes. We have conducted extensive evaluations which show that the proposed protocol not only improves the throughput and delay performance greatly, but also benefits the fairness.

I. Introduction

Ocean exploitation started from decades ago [**?**], however, newer and more efficient methods to explore the ocean, such as underwater acoustic sensor networks (UWASNs) [**?**], only emerged recently. UWASNs play an important role in ocean applications, such as oceanographic data collection, pollution monitoring, offshore exploration, disaster prevention, assisted navigation and so on. While terrestrial sensor networks (TWSNs) are densely deployed, in underwater, the deployment is deemed to be more sparse, due to the high cost of underwater sensors [**?**]. UWASNs employ sound to perform wireless communications in the ocean because of its low attenuation property in the water [**?**]. In comparison, TWSNs use radio-frequency electromagnetic wave to communicate in the air. The speed of sound in the water is 1.5km/s while that of radio waves in the air is 300,000km/s. As a result, poor throughput and high delay become the bottleneck of UWASNs.

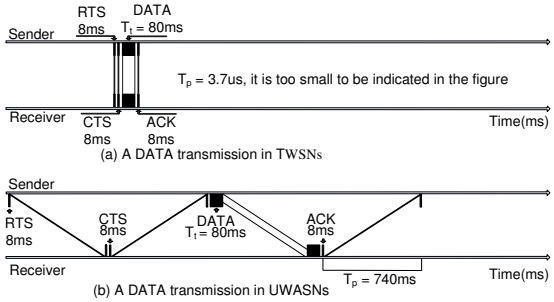

Fig. 1. A data transaction process.

TABLE I. Parameters for data transmissions.

Parameter	Value
Data Rate	10 kbps
Data Packet Size	100 bytes
Control Packet Size	10 bytes
Transmission Range (communication range)	1500 m
Interference Range	3500 m
Average Distance between Two Nodes	1110 m

As an example, Fig. 1 demonstrates the bottleneck of UWASNs by comparing with TWSNs in a data transaction process. The data transaction of UWASNs takes much longer time than that of TWSNs, and thus the throughput is much lower. In Fig. 1, parameters [**?**] in Table I are adopted, and both TWSNs and UWASNs employ the CSMA/CA mechanism [**?**] to transmit a packet. Then we

TABLE II. T_t and T_p for TWSNs and UWASNs.

	T_t	T_p
TWSNs	80ms	3.7us
UWASNs	80ms	740ms

calculate the data transmission time (T_t) and the propagation time (T_p) for TWSNs and UWASNs. The results are shown in Table II, in which $T_p \ll T_t$ for TWSNs, and $T_t \ll T_p$ for UWASNs. The long T_p of UWASNs (740ms) amplify the throughput and delay penalty of handshaking protocols [**?**], thus UWASNs should not apply the CSMA/CA as TWSNs do.

Although the long propagation time T_p results in bottleneck in UWASNs, it enables parallel data transmissions as long as there is no collision. It enables the start of another data transaction before the end of current data transaction. In this way, we can mitigate the bottleneck with priority scheduling protocol called RAS (application based scheduling protocol). We summarize our contributions as follows:

(1) After distinguishing the scheduling problem of UWASNs with that of TWSNs, we formulate the scheduling problem. Then we prove that the complexity of the algorithm to solve such a problem is exponential.

(2) By parallel transmissions and utilizing the information from routing and application layer, we design an efficient priority scheduling protocol at the MAC layer of the base station (BS).

(3) Extensive evaluations are conducted to show that the proposed protocol not only improves the throughput and delay performance in multi-hop networks, but also enhances the fairness.

In the remainder of this paper, Section II describes related work. Section III introduces the network scenario and the scheduling problem. Section IV formulates and analyzes the scheduling problem for UWASNs, then Section V illustrates RAS. Section VI evaluates the performance of the proposed RAS approach, and Section VII concludes the paper.

II. Related Work

Recently, there are extensive research efforts focusing on improving the performance of UWASNs by enhancing the traditional data transmission model at the MAC layer. Slotted FAMA [**?**] is a handshaking-based protocol designed to improve the traditional data transmission model and avoid collisions caused by the hidden terminal problem. It synchronizes all nodes and makes all the transmissions start at the beginning of a slot. However, since the slot length is too long, its throughput and delay performance improvement is not obvious.

The reservation MAC protocol proposed in [**?**] increases the throughput by separating two channels: a control channel for RTS/CTS handshake and a data channel for data transmission. Nevertheless, the protocol requires the nodes to be equipped with two sets of transceivers. Also, the work only simulates a simple scenario with one-hop communications. On the other hand, our work can be applied in either one-hop or multi-hop topologies, and does not need another set of transceiver.

Regarding the RTS/CTS handshaking transmission model as inefficient in UWASNs, Chirdchoo et al. propose two aloha-based MAC protocols [**?**] and Affan et al. develop T-Lohi [**?**] which uses short wake-up tones to reserve channel for data transmission. UW-FLASHR [**?**] is based on TDMA rather than RTS/CTS handshaking. It divides its cycle into two portions: experimental portion and DATA portion. In the experimental portion, control frames RTS and CTS for requesting new transmission time slots within the DATA portion are exchanged. In the DATA portion, nodes only transmit data in the already acquired time slots. Although our method is also based on TDMA, our scheduling elements are different from the existing work.

III. Overview

There are three practical network scenarios for UWASNs: static two-dimensional UWASNs for ocean bottom monitoring, static three-dimensional UWASNs for ocean column monitoring, and three-dimensional networks of autonomous underwater vehicles [**?**]. In this paper, we focus on the first scenario and address its scheduling problem in a BS-centered multi-hop topology.

A. Network Scenario

In the network scenario, the typical application is the surveillance application, in which all nodes generate the same amount of data and send them to the BS periodically with cycle T_c. Supposed that the UWASNs perform synchronization with the existing techniques [**?**], so that the nodes can work and sleep periodically [**?**]. Cycle T_c is divided into two portions. One portion is the sleeping period, the other is the working period T_w which is divided into n time slots T_s. The length of one *time slot* is equal to the transmission time of a data packet T_t plus guard time T_g, i.e., $T_s = T_t + T_g$. If there is a data burst due to abnormal events, nodes can transmit them in the following sleeping period by notifying the related nodes in advance. In this way, data burst does not require updates of the working schedule. In the future we focus on analyzing the working schedule in one cycle.

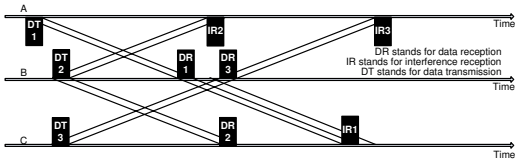

Fig. 2. Data transmissions among 3 nodes in UWASNs.

Since the sensor nodes are anchored to the bottom of the ocean and are thus static. It is practical to employ static routing. Also, the deployment of underwater sensors is very sparse compared with the dense sensors in TWSNs. The total number of nodes in a UWASN is usually less than 100. Thus it is reasonable to assume that the BS knows the position of every node, and it can calculate the expected number of data that each node has to send and receive in a cycle with static routing. Finally, the BS can calculate the working schedule for all the sensor nodes in UWASNs. These processes cost little efforts because they do not require frequent updates in this scenario. Therefore, the major challenge is how to make the working period of the whole network as short as possible so as to save the energy and prolong the network lifetime.

B. Scheduling Element

Next, we show that the scheduling problem of UWASNs is different from that of TWSNs because the *scheduling elements* of the former is more complex than that of the latter. Since the purpose of scheduling is to arrange the data transactions of all nodes, the scheduling element corresponds to one data transaction. It consists of three time points: data transmission (DT) time, data reception[1] (DR) time, and a sequence of interference reception [2] (IR) time. For example, in Fig. 2, the scheduling element of node A's first data transmission to B is composed of DT1, DR1 and IR1. They occur at different time because T_{p} in UWASNs is long enough (740ms) to distinguish them. In contrast, the three time points are the same in TWSNs because T_{p} in TWSNs is too small (3.7us) to differentiate them [**?**]. Therefore, the scheduling element of TWSNs is simpler than that of UWASNs.

Due to $T_{\mathrm{t}} \ll T_{\mathrm{p}}$ in UWASNs, we can improve throughput and delay performance by parallel transmission. Actually, nodes should transmit or receive at any time as long as there are no collisions. In this way, channel idling is avoided, thus throughput and delay performance can be enhanced. For example, in Fig. 2, nodes B and C transmit packet 2 and packet 3 to each other simultaneously, but their receptions are not collided. In this case, collisions will surely happen in TWSNs.

[1]reception of a packet destined for it.

[2]reception of a packet not destined for it.

TABLE III. Notations.

$\mathbb{S}$	the set containing all the nodes, including the BS.
K_m	node m's children number.
C_{mj}	node m's j-th child, $j \in \{1, 2, \cdots, K_m\}$.
$DT_{C_{mj}}$	the time node C_{mj} transmits a packet to its parent m.
$DR_{C_{mj}}$	the time parent m receives a packet from its child C_{mj}.
$\mathbb{S}_{C_{mj}}$	the set of nodes within node C_{mj}'s interference range.
$IR_{SC_{mj}}$	the time each node in set $\mathbb{S}_{C_{mj}}$ receives interference packets from node C_{mj}.
Q_{jw}	the w-th packet that child C_{mj} sends to its parent m.
W_{mj}	the number of packets child C_{mj} has to transmit to its parent node m.
$R_{C_{mj}Q_{jw}}$	the time node m receives each of the packets from its children.
L_1	the optimal schedule length, i.e., the solution of the scheduling problem.
L_2	the schedule length calculated with RAS.
L_3	lower bound schedule length.

IV. The Scheduling Problem in UWASNs

In this section, we first introduce the scheduling principles, which are the constraints for the following scheduling problem formulation. Table III lists some notations used in Section IV and Section V.

A. Scheduling Principles

The transceiver cannot receive when it is transmitting, and collision will occur at a node when it receives more than one packet [**?**][3]. In order to avoid collision, we define the following scheduling principles:

(1) A DR duration must not overlap any DT duration.

(2) A DR duration must not overlap any IR duration.

(3) A DR duration must not overlap any other DR duration.

(4) A DT duration and IR duration(s) can overlap.

Since there is no data from the BS to sensor nodes, we can design a compact schedule by only allocating time for data from sensor nodes to the BS. Thus the next principle is:

(5) No DR from i-th hop node to $(i+1)$-th hop node.

The goal of data transactions is to guarantee successful receptions, and we arrive at the last principle:

(6) A node considers DR duration as the scheduling basis rather than DT or IR duration.

[3]This corruption is called interference. Interference packets at a node are divided into two types: the first type is for packets that are not destined for the node, but are within the node's communication range RR. The other type is for packets that are beyond the node's communication range, but are within the interference range RI. Usually, the relation [**?**] between RR and RI is: $2 \times RR \leq RI \leq 3 \times RR$.

With DR as the scheduling basis, we do not have to consider whether IR will overlap other IRs and DTs. In addition, we can increase the throughput and reduce delay by making nodes transmit or receive instead of idling whenever no DR is overlapped.

B. Scheduling Problem Formulation

Under parallel transmissions, we aim at calculating the shortest working period of a cycle within which all the nodes' expected data are transmitted and received, such that we can make the nodes sleep as the long as possible while finishing their communication. We formulate this problem as follows.

Let $\mathbb{S}$ be the set containing all nodes, including the BS. Suppose node m has K_m nodes that send packets to it, so that node m is called a *parent* and the K_m nodes are called its *children*. Each child of m is denoted by C_{mj}. Node m has K_m types of elements $E_{C_{mj}}$, which is composed of three time points as follows:

$$E_{C_{mj}} = \begin{cases} DT_{C_{mj}} \\ DR_{C_{mj}} \\ IR_{SC_{mj}} \end{cases} \quad (1)$$

In Equation (1), the notation of each component is explained in Table III.

There is a relationship among the three components of $E_{C_{mj}}$, $\forall m \in \mathbb{S}$, $\forall j \in \{1, 2, \cdots, K_m\}$. If $DR_{C_{mj}}$ is known, then the other two times could be determined. Therefore, we only need to calculate the time a node $m \in \mathbb{S}$ receives each of the packets from its children. We denote this time as $R_{C_{mj}Q_{jw}}$, where Q_{jw} and W_{mj} are denoted in Table III. The time for node m to finish all the receptions from its children is $\max R_{C_{mj}Q_{jw}}$, $\forall j \in \{1, 2, ..., K_m\}$, $\forall w \in \{1, 2, ..., W_{mj}\}$. Since our objective is to minimize the working period of the whole network in a cycle, we attempt to find a schedule so that all nodes finish all their receptions as early as possible. We formulate our problem as follows:

$\min\max R_{C_{mj}Q_{jw}}$,
$\forall j \in \{1, \cdots, K_m\}$, $\forall j' \in \{1, \cdots, K_{m'}\}$, $\forall w \in \{1, \cdots, W_{mj}\}$, $\forall w' \in \{1, \cdots, W_{mj'}\}$, $\forall m \in \mathbb{S}$.

$$\text{s.t.} \begin{cases} R_{C_{mj}Q_{jw}} > R_{C_{zj'}Q_{j'w'}} + DT_{C_{zj'}} + T_{\mathrm{t}}, \\ \quad \text{or } R_{C_{mj}Q_{jw}} + T_{\mathrm{t}} < R_{C_{zj'}Q_{j'w'}} + \\ \quad DT_{C_{zj'}}, \forall z \in \{z | C_{zj'} = m\}, & (2) \\ R_{C_{mj}Q_{jw}} > R_{C_{zj}Q_{j'w'}} + IR_m + T_{\mathrm{t}}, \text{ or} \\ \quad R_{C_{mj}Q_{jw}} + T_{\mathrm{t}} < R_{C_{zj'}Q_{j'w'}} + IR_m, \\ \quad \forall z \in \{z | m \in \mathbb{S}_{C_{zj'}}\}, & (3) \\ R_{C_{mj}Q_{jw}} > R_{C_{mj'}Q_{j'w'}} + T_t, \text{ or} \\ \quad R_{C_{mj}Q_{jw}} + T_t < R_{C_{mj'}Q_{j'w'}}, \forall j \neq j', & (4) \\ R_{C_{mj}Q_{jw}} + DT_{C_{mj}} \geq 0. & (5) \end{cases}$$

T_{t} in the above inequalities is the duration for one data transmission. Inequalities (2)-(4) are constrained by our scheduling principles (1)-(3), and Inequality (5) means that the transmission time of a packet should be equal to or greater than 0.

C. Scheduling Problem Analysis

We call the solution of the problem as the *optimal schedule length* L_1. Unfortunately, although L_1 is optimal, it is impossible to be calculated in limited time. The reasons are: Firstly, the problem is not a linear programming problem as the objective function $\min\max R_{C_{mj}Q_{jw}}$ is not linear; Secondly, even though the objective function can be changed to be linear with some mathematical transformations, the complexity is still exponential due to the constraints.

Considering the "or" argument in the constraints, if the constraints contain one "or" argument, then the problem is divided into two subproblems. Likewise, if "or" appears n times, then the above problem is divided into 2^n number of subproblems. For example, in a network with N nodes, each node is constrained by Inequalities (2), then the total number of inequality (2) is at least N, i.e., the number of "or" argument is N. Thus the problem is at least divided into 2^N subproblems. As a result, the complexity of the algorithm for the above problem is exponential.

Actually, we implement the algorithm in a Linux server with a CPU as Intel Core Duo 2.8GHz and 4GB RAM. It takes one day to calculate the shortest schedule for a 9-node network. It takes about three days for a 16-node network. However, it will never end in two weeks for a 25-node network.

V. RAS Protocol

In this section, we propose an efficient scheduling algorithm, a routing and application based scheduling protocol RAS, to solve the formulated scheduling problem with an near-optimal solution. According to the network scenario in the overview section, it is practical for RAS to take advantage of the routing information and only schedule data transmission from sensor nodes to the BS. With them, RAS can calculate a compact schedule for data exchange under the same scheduling principles.

A. Scheduling Algorithm of RAS

We call the schedule length calculated with RAS as L_2, and obviously $L_1 \leq L_2$. In RAS, the scheduling elements are shown in Equation (1), and a transaction of data transmission and reception will last for several time slots. According to scheduling principle (6), we will

schedule all the elements generated in a cycle equals by scheduling all the data receptions in a cycle. Let $\mathbb{S}_i = \{m :$ node m is i hops from the BS, $m \in \mathbb{S}\}$. Assuming a node's distance from the BS is proportional to its hop distance to the BS, the following is the priority scheduling steps of RAS algorithm:

Step1: Schedule the BS's DR from 1-hop nodes.

Step2: Schedule the DR tier by tier: from inner tier to outer tier, i.e., from DR of nodes in $\mathbb{S}_i$ to DR of nodes in $\mathbb{S}_{i+1}$, $i \in \{1, 2, 3, \cdots, H-1\}$, H is the maximum hop distance to the BS.

Step3: For each node $m \in \mathbb{S}_i$ that is going to receive data packets from its children $C_{mj} \in \mathbb{S}_{i+1}$, $j = 1, 2, ..., K_m$, arrange its DR from its children alternatively. For example, node 1 has two children A and B. In a cycle, each of them send 3 packets P_{Ai}, P_{Bi} to node 1, and $i \in 1, 2, 3$. Then one possible DR sequence at node 1 is: $P_{A1}, P_{B1}, P_{A2}, P_{B2}, P_{A3}, P_{B3}$.

Algorithm 1 CalcSchedule() function at the BS.

```
1:  Parent = BS; hop = 1. //schedule the BS's DR from 1-hop nodes
2:  while hop ≤ maxhop. do
3:    while Parent has children. do
4:      while Parent has data to receive from its children. do
5:        if Parent is idle in the Time Slot Slot. then
6:          With global information, Parent searches its entire children to alternatively find a child whose transmission results in its reception at the Slot.
7:          if Parent finds a suitable child. then
8:            schedule the child's transmission and the related reception and interference.
9:            break searching.
10:         end if
11:       end if
12:       Parent fetches the next Slot for reception.
13:     end while
14:     fetch the next Parent to schedule reception.
15:   end while
16:   hop = hop + 1. //schedule the DR tier by tier
17: end while
```

The reason to schedule the BS's DR first is that the topmost goal is for the BS to receive all the data generated by all other nodes in a cycle. The reason to prioritize the inner tier nodes over the outer tier nodes is that the inner-tier nodes are affording much heavier traffic. It is unfair for them to share the same bandwidth with other light-traffic nodes. In addition, the packets forwarded by the inner tier nodes are forwarded more hops than those forwarded by the outer tier nodes. Colliding or dropping those packets would cost more efforts to retransmit. Finally, to alternate the receptions among the children provides load balancing of the nodes. If fairness is not considered for a parent's children, a few children might drop packets due to congestion whereas other children's queues are far from full. Therefore, when we alternate the children's transmissions, we improve the fairness. The three steps are shown in Algorithm 1.

B. Analysis of the Scheduling Algorithm

Since the upper bound for the RAS schedule length (L_2) can be infinitely long, we only discuss L_3, the lower bound for schedule length. Assuming each node generates P packets in a cycle, then the BS has to receive $N \times P$ packets in total from 1-hop nodes in a N-node network. To receive packets, it has to wait for at least the propagation time T_{p} of one packet. Therefore, the shortest time for receiving all the $N \times P$ packets is $N \times P \times T_s + T_{\mathrm{p}}$. We call this time L_3 as the *lower bound*, which cannot be achieved in large-scale networks due to interferences. L_3 is the shortest time for the BS to finish all its receptions while L_1 is the shortest time for all the nodes in a network to finish their receptions. Therefore, L_3 is no larger than L_1. Besides, $L_1 \leq L_2$, then $L_3 \leq L_1 \leq L_2$. We can use L_3 to indicate the lower bound for L_2 instead of using L_2 which is intractable.

VI. Performance Evaluation

We conduct the performance evaluation in the popular freeware network simulator ns-2 [**?**] with parameters shown in Table I. Using the setdest tool of ns-2, we generate network scenarios of the same node density with six different sizes: from 9-node network to 64-node network. In those networks, nodes are randomly distributed and connected without holes, and the maximum hop distance ranges from 1 hop to 7 hops. For networks of each size, we calculate the performance under about 10 different topologies and show the average results. We employ the underlying and traditional propagation model of ns-2 after adjusting the transmission medium parameters as [**?**] and [**?**] do. The reason of using such propagation model is that although acoustic waves in water suffer significant absorptive losses, scattering and refraction, those factors do not change the latency relationship among DT, DR and IR.

In the following we first compare the schedule length of RAS with the lower bound schedule length. Next we implement RAS and UW-FLASHR [**?**] in ns-2, and compare their performance in terms of throughput, delay and fairness.

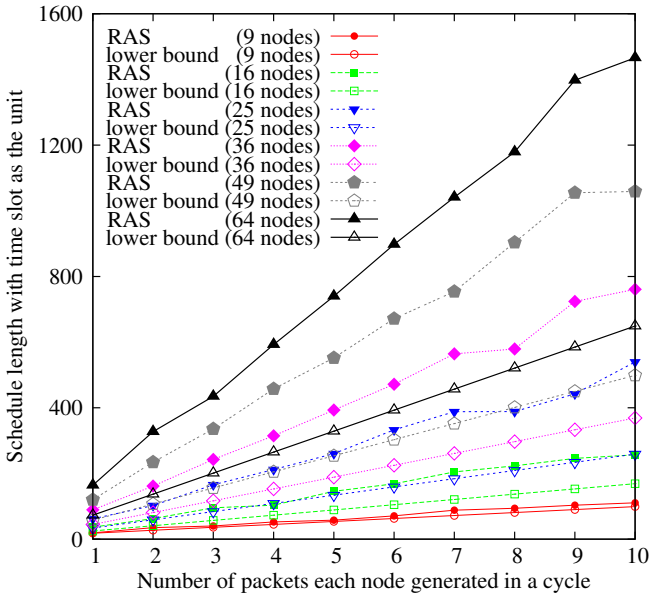

Fig. 3. RAS schedule length vs. lower bound schedule length.

A. Schedule Length

In this subsection, we calculate the lower bound schedule length L_3 and the RAS schedule length L_2 when the number of packets generated by each node in a cycle varies from 1 to 10. We do not draw the lines for the optimal schedule length L_1 because it is intractable in limited time. In addition, since $L_3 \leq L_1 \leq L_2$, if L_2 is close to L_3, then it closer to L_1.

In Fig. 3, all the lines are almost linear, which means that RAS is scalable in calculating the schedule no matter the traffic rate is low or high. Since the schedule length is the working time under a certain traffic rate, the shorter it is, the longer the time that the whole network could sleep for. The whole network could increase their throughput by working in the sleeping period. Therefore, the shorter the schedule length is, the higher the throughput could be. Observing the value of L_3/L_2, it stabilizes at around 0.5 in networks with node number larger than 16. which means that RAS's throughput can achieve about 50% of the lower bound. Since $L_3 < L_1$, thus RAS's throughput must be more than 50% of the optimal case. It is much better compared with T-Lohi [**?**] whose throughput is about 30% of L_1. However, the ratios of 9-node and 16-node networks are much higher than 50%. In this case, the schedule calculated by RAS is very close to the lower bound. The reason is that the above two networks are small-scale networks, in which the hop distances are 1-hop or 2-hop. Therefore, they suffer less from the interferences caused by neighboring nodes. When there are few interferences, the schedule length is reduced. That is why L_2 of small-scale networks are much closer to L_3 than large-scale networks.

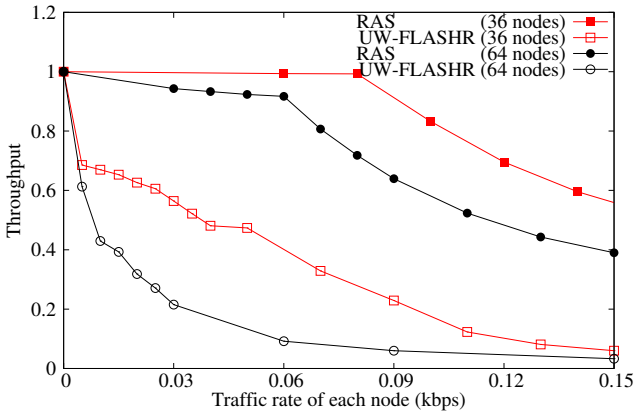

Fig. 4. Throughput comparison

B. Network Throughput

Due to RAS's high scalability in schedule length demonstrated in the previous subsection, we use the schedule calculated for the case when only one packet is generated in a cycle. To compare RAS with UW-FLASHR [**?**], an existing MAC protocol designed for high channel utilization, we employ their throughput definition. Throughput is defined by measuring the total number of the intended data packets received by the BS by the total number of data packets generated by all the nodes in a period. Obviously, if the traffic generated at each node is so heavy that it exceeds the maximum capacity of the network, then the throughput would drop, and even approaching to 0. Conversely, if the traffic is light, then it is likely that all the data generated will be received by the BS; therefore, the throughput is 1 when there is no traffic.

Although we simulated networks of different sizes, for the sake of conciseness, Fig. 4 only compares the throughput of RAS and UW-FLASHR in 36-node networks and 64-node networks. As the traffic rate increases, the throughput of all the networks drops from 1. In addition, 36-node networks are able to afford a much heavier traffic rate than the 64-node networks because networks with a larger size suffer higher total traffic. Moreover, we notice that the throughput for 36- and 64-node networks with UW-FLASHR dramatically drops from 1 when the traffic rate is not 0. This is because UW-FLASHR performs the slot requirement among the neighbors, and the hidden terminal problem leads to some slot establishment which might cause collisions. On the other hand, RAS performs scheduling based on all nodes' position information, thus no collision happens. Finally, for UW-FLASHR, the heav-

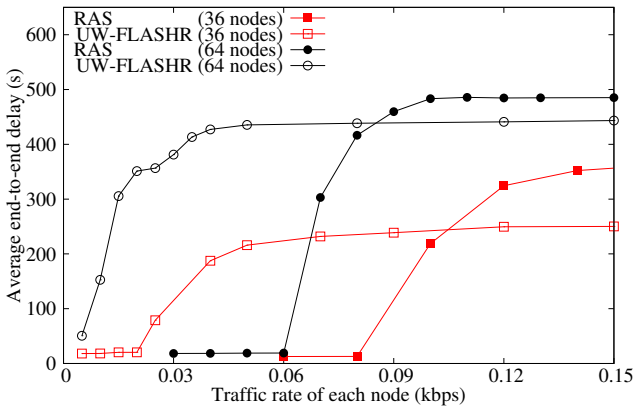

Fig. 5. End-to-end delay comparison

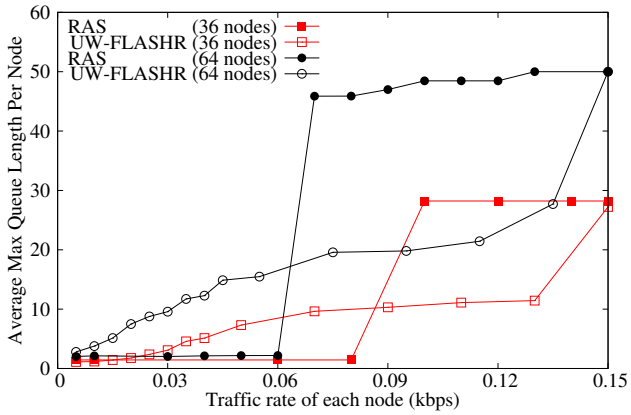

Fig. 6. Queue length comparison.

iest traffic rate these networks could afford is much less than that of RAS. This is because RAS generates a much compacter schedule than UW-FLASHR. RAS arranges the exact time needed by the transmission and reception of each node while UW-FLASHR reserves the time slots for transmission randomly.

C. Average End-to-end Delay

The end-to-end delay is the period from the time a packet is generated by a node until the time it is received by the BS. Fig. 5 shows that the average end-to-end delay increases when the traffic rate increases. Specifically, when the traffic rate is heavy enough to cause congestion in the networks, there are sudden jumps of delay as observed in the figure. By observing the sudden jumps in delay, we find that RAS networks can afford about 4 times higher traffic load than UW-FLASHR networks without collision. Because the scale of 36-node networks is smaller than 64-node networks, their end-to-end delay is also shorter. In addition, when heavily congested, the delay of RAS networks and UW-FLASHR networks stops increasing with traffic rate. The reason is that both RAS and UW-FLASHR are based on TDMA to reserve the channel rather than on CSMA/CA to compete the channel, thus the delay reaches an upper bound. However, the delay upper bound of RAS networks is higher than that of UW-FLASHR networks, this is due to that: the priority scheduling makes the queue utilization of RAS networks higher than that of UW-FLASHR networks (this will be explained in the following subsection). In UW-FLASHR networks, the queue utilization is very low. Most packets from faraway nodes cannot arrive at the BS before being dropped by the heavy-loaded forwarding nodes. In other words, most of packets that arrive at the BS are generated by nearby nodes, thus the delay upper bound is lower. In contrast, this phenomenon is alleviated by the priority scheduling in RAS networks.

D. Average Maximum Queue Length per Node

In this subsection, we demonstrate the advantage of RAS in fairness by showing that its queue utilization is fairer than that UW-FLASHR. The queue size of each node is set to 50 in the simulations. If a queue is filled with 50 packets, then further packet arrival will cause one packet to be dropped.

UW-FLASHR does not take the application direction into consideration, nor does it arrange longer time for nodes with heavier traffic. As a result, nodes with heavier traffic (i.e., nodes that are nearer to the BS) would easily accumulate a long queue of packets and suffer queue overflow very soon while nodes with lighter traffic maintain an empty queue. As a result, the queue utilization of the nodes in UW-FLASHR is very unfair and low. Actually, nodes with heavier traffic experience a larger packet arrival rate, and they need more time to handle the packets. RAS gives higher priority to nodes with heavier traffic by allocating more data transmission time to them, thus their packet leaving rate is also higher. Likewise, nodes with lighter traffic is allotted less time. As a result, the queues of all the nodes are balanced, and the queue utilization is fairer.

Due to similar phenomenon of 36-node networks and 64-node networks, we mainly discuss the case for 64-node networks in Fig. 6. When the traffic rate is between 0kbps to 0.06kbps, the queue length of RAS networks is shorter than that of UW-FLASHR networks. Due to RAS's capability of affording higher traffic rate, most of the nodes in RAS networks does not have to queue under those traffic rates. Whereas the nodes in UW-FLASHR networks are queueing more and more packets when traffic gets heavy. In addition, RAS networks undergo a jump in queue length when the traffic rate increases from 0.06kbps

to 0.07kbps. After this the queue length of RAS networks is much higher than that of UW-FLASHR networks until they both reach the upper bound 50. This because the queue utilization of RAS networks is fairer and higher than UW-FLASHR networks. At traffic rate 0.07kbps, RAS networks start to congest, most of the nodes suffer and their queues overflow. In contrast, in UW-FLASHR networks, the queues of the few nodes that are next to the BS are congested at a very low traffic rate while the queues of faraway nodes are empty. Other nodes get congested gradually with the increasing traffic rate, thus the queue length does not surge.

Furthermore, because small-scale networks are less likely to suffer congestion, the queue length of 36-node RAS networks soars at about traffic rate 0.09kbps rather than at 0.07kbps. It stabilizes at around 27 rather than at 50 when the traffic rate is larger than 0.11kbps. Eventually, it will stabilize at 50 when the traffic rate is very high. Nevertheless, 0.15kbps is not a very high traffic rate for 36-node networks, thus only a majority of the nodes are congested while the others sustain empty queue. Still the queue length of RAS networks is much larger than that of UW-FLASHR networks when congestion happens in RAS networks, which again indicates that the queue utilization of RAS networks is fairer and higher.

In summary, greater maximum queue length allows fairer and higher utilization of the queue. Correspondingly, the delay upper bound of the RAS networks is higher than the UW-FLASHR networks because more faraway packets are capable of arriving at the BS with no overflow in queue.

VII. Conclusions

In this paper we propose a novel priority-based scheduling algorithm to mitigate the communication bottleneck of UWASNs. The different characteristics of communication and node scheduling between UWASNs and traditional TWSNs are investigated. We then formulate the scheduling problem for UWSANs and analyze its complexity. We provide a heuristic algorithm to solve the scheduling problem with a newly designed scheduling-based MAC protocol, called RAS. RAS gives higher priority to nodes with heavier loads. We compare RAS with two state-of-the-art approaches T-Lohi and UW-FlASHR. The simulation results demonstrate that RAS not only effectively improves the throughput and delay, but also increases the queue utilization and fairness. In the future, we are interested in applying RAS for real deployment and investigating its performance in terms of delay and throughput bottleneck in practice.