Telecommunication Systems
https://doi.org/10.1007/s11235-021-00812-y

# An SDN-based framework for QoS routing in internet of underwater things

**Reza Mohammadi**[1] · **Amin Nazari**[1] · **Mohammad Nassiri**[1] · **Mauro Conti**[2]

Accepted: 12 June 2021


**Abstract**
In recent years, the Underwater Internet of Things (IoUT) has become a popular technology for exploring the underwater environment. IoUT enables administrators to explore and monitor underwater environmental phenomena from anywhere in the world where there is Internet access. Due to the harsh underwater environment, the reliability of communication between sensor nodes deteriorates, causing certain performance issues such as higher packet loss rate and long end-to-end delay. Therefore, it is essential to manage the communications between the sensors to address these problems in order to improve the QoS. Software-defined networking (SDN) is one of the most promising architectures for providing efficient network management by decoupling the data plane from the control plane of the network. This paper proposes a new QoS routing technique for SDN-based IoUT aiming at improving QoS by establishing reliable paths between sensor nodes. To do this, the controller gathers the 3D coordinates of each underwater sensor in order to compute the distance between the nodes. Then, it estimates the reliability of each link by using underwater acoustic equations. Finally, it calculates the most reliable path with minimum delay and installs the path on the nodes located along it. The experimental results show that our mechanism significantly outperforms other non-SDN approaches in terms of several performance measures ranging from packet loss ratio and end-to-end delay to energy consumption.

**Keywords** IoT · IoUT · Underwater networks · QoS · SDN

## 1 Introduction

The earth is a planet covered with more than 70% of water. Therefore, there are many unexplored underwater areas. Since 2000, the advent of underwater wireless sensor networks (UWSN) has enabled not only the exploration of unknown underwater areas, but also many possible underwater applications [1–3]. In a nutshell, an UWSN consists of a number of sensor nodes which are distributed in an underwater area to collect the desired information with respect to its application. Some interesting applications are environmental monitoring, disaster prevention, mine reconnaissance, distributed tactical surveillance, and so forth [4].

Since the electromagnetic signal faces a strong attenuation in the underwater environment, underwater sensors use acoustic signals as a means of transmission [5]. Unfortunately, the acoustic signals have a low bandwidth and a long propagation delay [4]. In addition, the speed of sound under the water depends on the salinity, temperature, and pressure of the water [6]. All these limitations lead to a lack of reliability of the submarine links. Moreover, the UWSN suffers from certain challenges such as energy constraint [7,8], high channel impairment, and high packet loss [9,10]. Besides, since replacing the battery is not possible in most cases for underwater sensors, energy efficiency becomes another important problem in such networks.

Recently, the Underwater Internet of Things (IoUT) has been proposed as a new technology that intends to connect the UWSN to the Internet in order for people to facilitate

✉ Reza Mohammadi
R.mohammadi@basu.ac.ir

Amin Nazari
aminnazari91@gmail.com

Mohammad Nassiri
m.nassiri@basu.ac.ir

Mauro Conti
conti@math.unipd.it

[1] Computer Department, Faculty of Engineering, Bu-Ali Sina University, Av. sh. Fahmideh, Hamedan, Iran

[2] Department of Mathematics, University of Padova, Padova, Italy

Published online: 24 June 2021

controlling and monitoring their underwater networks from remote terrestrial locations [11]. Indeed, IoUT, first introduced as a world wide network of smart underwater objects which are connected together and can be monitored from remote sites [12]. Each of smart objects has different roles in an underwater environment [11]. Sensor nodes are used to observe environmental phenomena as well as relaying data packets from source to destination. Autonomous Underwater Vehicles (AUVs) move around underwater sensors and send them some controlling commands or collect their observed data in order to deliver them to a sink node. It is worth to mention that, IoUT is more similar to terrestrial-based counterpart (IoT) in terms of structure, functionalities, limitations in energy and computation and so on [12]. In IoUT, usually, the sink node is located on the water surface and equipped with acoustic and electromagnetic transceivers. In fact, it communicates with sensor nodes and AUV by using acoustic signals and also communicates with onshore station or satellite by using electromagnetic signals. The role of the earth station is to receive the data collected from the sink, to analyze or summarize it and finally to send the results to the monitoring center via the Internet infrastructure. Satellite links can help forward data from the sink to the onshore station in situations where the sink node and the onshore station are in far distance from each other. As shown in Fig. 1, each sensor can sense various types of phenomena such as pressure, temperature or chemicals, then forward them to the sink. A major limitation in such architecture is that, each sensor has no overall knowledge of the network and the current conditions of the underwater channel. More precisely, each sensor only knows its neighbors. Moreover, due to the lack of global knowledge of the network, routing protocols cannot establish an optimal route between a pair of sensors or between a sensor and the sink. This limitation might cause to transmitting data over an unreliable path with a long delay.

Software Defined Networking (SDN) is a new networking paradigm which decouples the control plane from the data plane [13]. This separation facilitates a dynamic and fine-grained network management and also allows the provisioning of quality of service (QoS) [14]. The controller module has a global knowledge of the network. It collects valuable information from data plane equipment and also manages them by sending control commands. The data plane devices follow the commands upon receiving a packet. In short, the controller determines how the network devices should behave. The concept of SDN can be used for UWSNs to overcome certain limitations such as interoperability issues between heterogeneous underwater devices from different vendors [1], efficient network management and QoS provisioning [15].

In an SDN-based UWSN, each underwater device is programmed and configured by a centralized controller [1,16]. For instance, OF-sensor nodes are programmable underwater nodes which are able to communicate with the controller [17]. As shown in Fig. 1b, the controller in an SDN-based UWSN has a global knowledge about the network devices such as AUV or sensors and thus can manage the network in an efficient manner. It can install appropriate forwarding rules on sensor nodes with respect to the current network condition.

In this article, we propose a new SDN-enabled architecture that facilitates the installation of reliable paths between network equipments in order to improve QoS provisioning.

In our proposed architecture, at the first step, underwater nodes notify the network controller (Sink) of their coordinates. After collecting the coordinates of each node, the controller computes the Euclidean distance between the nodes. In addition to distance, the controller uses underwater communication equations to calculate the reliability of the link between neighboring nodes. Knowing the length and reliability of each link in the network, helps the controller compute the suitable paths between source and receiver nodes in terms of delay and reliability. After calculating the paths, the controller installs it on all relevant nodes along the path between the source nodes and the sink on the sea surface. This behavior leads to improve QoS because the computed path has obtained using the global knowledge of controller, instead of using distributed routing protocols. We are not aware of any academic related work about leveraging the benefits of SDN in IoUT to yield more reliability and QoS. To the best of our knowledge, this paper is the first research work to achieve this goal. In summary, the contributions of this paper are three-fold:

- We propose a new SDN-enabled architecture for Internet of Underwater Things.
- We present and develop a centralized QoS routing schema to estimate the best path from each node to the sink.
- We conduct extensive simulations to evaluate the performance of our method and compare its results with those obtained from a none SDN-based architecture.

The remainder of this paper is organized as follows: In Sect. 2, some related works about SDN-based UWSN and IoUT are reviewed. The proposed architecture and problem formulation are described in Sect. 3, respectively. In Sect. 4, simulation details and test results are presented. Finally, conclusions are described in Sect. 5.

## 2 Related work

In this section, some related work on IoUT and SDN-based architecture for UWSN are surveyed.

The advent of IoT concept in 1985 [18] has led to a great revolution in all areas of computer networks includ-

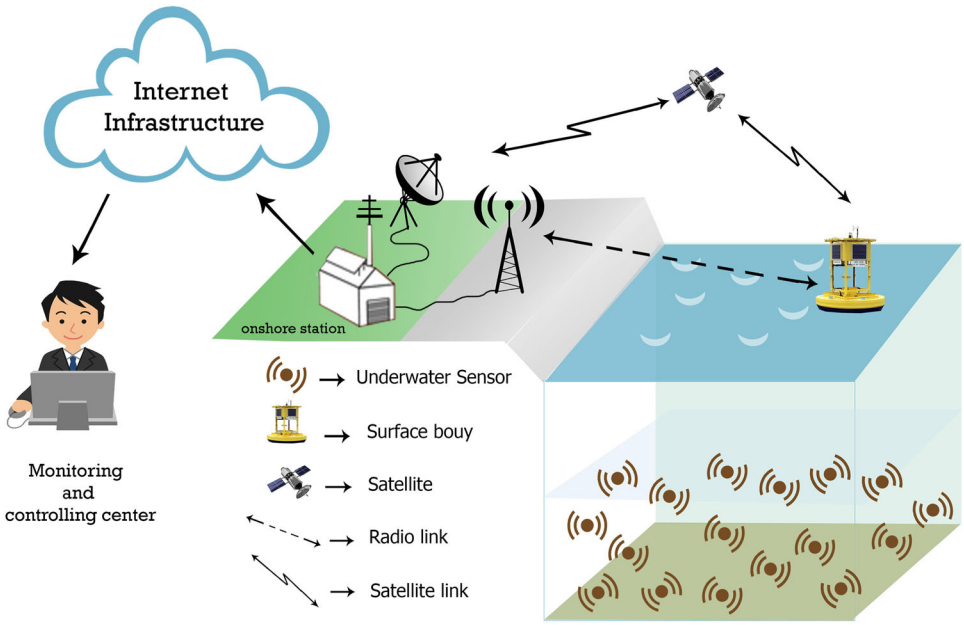



**Fig. 1** Sample architecture of IoUT

ing UWSN. Indeed, IoT enables to control or monitor a large number of interconnected smart objects such as physical devices and sensors from remote locations [19]. In light of this, IoUT was first introduced in 2012 by Domingo [12]. Berlian et al. [20] have proposed an IoUT-based framework for smart environment monitoring and analysis in real-time System. This framework first retrieves the obtained data from underwater sensors and Remotely Operated Vehicle (ROV). Example of these data could include total dissolve solid, pH, electrical conductivity, salinity, etc. Then, the data are analyzed in a data center platform based on Hadoop Multi-Node Cluster. Finally, the result of the analysis or the data collected is displayed in the form of a graph or table and can be accessed from anywhere in the world. Zhou et al. introduced Enhanced Channel-Aware Routing Protocol (E-CARP), a new routing protocol for underwater sensors [21]. E-CARP enables caching the collected data in the sink node and prevents forwarding the cached data over the network. This policy results in lower communication costs and an increase in network capacity.

Smart-IoUT has been proposed by Nayyar et al. [22] as a novel smart aquatic monitoring prototype. This prototype allows to access to the underwater live data such as temperature, pH, turbidity and dissolved Oxygen from world-wide. Urunov et al. [23] studied the key factor and problems of Network Management System (NMS) for IoUT. Moreover, they have presented the role of NMS in IoUT and proposed a topological discovery algorithm to improve management services in an underwater environment. In [24], authors have estimated the reliability of underwater links in IoUT in terms of Bit Error Rate (BER). To do this, they have investigated the impact of Signal-to-Noise Ratio (SNR) on BER. As a similar work [11], authors also attempted to estimate the probability of an underwater link in terms of successful delivery ratio.

Francisco et al. [25] have proposed Water Ping as a suppression protocol for ICMP and ICMPv6 for IoUT. The aim of Water Ping is to reduce the traffic and also enable some underwater applications such as monitoring lakes or oil pipelines. To achieve this goal, Water Ping only summarizes the header of ICMP without modifying the TCP/IP protocol stack and without the need for additional software or drivers. Xu and Liu [26] proposed an energy-efficient scheduling mechanism with high computation-utilization for IoUT , called EAST (Energy-Aware scheduling with Spatial-Temporal).Their mechanism is used to enable transmission of sensed data from the underwater objects. In order to eliminate exposed and hidden terminal problem, EAST uses probability-based contention model. Experimental results confirmed that EAST can outperform other representative underwater MAC protocols in terms of energy consumption, network throughput and success delivery ratio.

As mentioned in Sect. 1, in an UWSN or IoUT, the smart objects are spreaded in the underwater environment and they communicate with each other in a distributed manner. In such cases, due to the lack of a centralized management schema, the heterogeneity of devices and the harshness of underwater environment, network management faces significant limitations [1]. Fortunately, in last few years, some software defined technologies such as cognitive acoustic radio (CAR) [27], software defined radio (SDR) [28] and SDN [17] have been emerged to address the limitations of UWSN. However, software defined underwater acoustic networking, introduced by Torres et al. [29], was not a real SDN-based architecture. Their proposed approach was a

programmable acoustic modem named UANT(Underwater Acoustic Networking plaTform ) which uses GNU radio as a SDR framework. In [30], Torres et al. have discussed advantage of UANT and investigated some applications such as NetPerf, Ping and TinyOS tools using UANT. SoftWater is the first SDN-based architecture introduced by Akyildiz et al. [17] to provide scalability, energy efficiency and high network robustness and throughput. Moreover, it supports interoperability between underwater devices and enables various underwater applications.

Wang et al. have proposed an SDN-based solution in UWSN as an framework for big data [31]. The proposed solution maximizes network capacity while reducing management complexity. Qin et al. have proposed an Energy-aware Clustering protocol based on the Improved K-Means algorithm (ECBIK) [32]. They also introduced a novel Ant Colony algorithm combining with Markov Reward Process (R-ACO) to optimize the distance and angle of AUV path planning. Their simulation results showed that using R-ACO routing can increase node survival rates. In [33], Lin et al. have introduced DSR-SD as a routing schema for delay sensitive SDN-based UWSN. DSR-SD leverages SDN capabilities to gather network topology statistics and then estimates spatio-temporal characteristics. Finally, it calculates the routes and installs forwarding rules on corresponding intermediate nodes. Simulation results have shown that DSR-SD decreases the packet sojourn time in delay-sensitive underwater applications.

In [34], Luo et al. proposed a novel test-bed for SDN-based UWSN in which an RF wireless node communicates with the on-shore SDN controller using one-hop out-of-band RF channel. In this test-bed, underwater sensors use acoustic signals for communication and serve as the data plan. Fan et al. presented a centralized approach as the network control plane to manage and control an AUV network [35]. The role of the control plane in their approach was to plan routing and movement decisions for each individual AUV. Alostad proposed a dynamic adaptive routing (DAR) protocol for teh IoUT environment to improve the reliability of packet transmission. The main objective of DAR is to maintain the trade-off between reliability and network lifetime. To achieve this goal, it calculates routes by using an optimal directed acyclic graph (DAG). The simulation results showed that DAR improves the packet delivery ratio in underwater environments where the Bit-Error-Rate (BER) is high.

In order to compare the above research work in terms of some influential parameters in underwater environments such as mobility, energy, delay, and reliability, we summarize them in Table 1. As shown in this table, some of the research works focus on realizing IoUT concept while others have exploited SDN paradigm in UWSN to reduce management complexities. However, none of them has been investigated using SDN in IoUT specially for providing better reliability and QoS. More precisely, to the best of our knowledge, no existing works focus on the role of SDN architecture in IoUT. To fill this gap, in this paper, we intend to use SDN as a centralized management schema for IoUT in order to achieve better performance in terms of end-to-end delay, reliability and energy consumption.

## 3 Proposed SDN-based IoUT architecture

In this section, first we introduce our SDN-based IoUT architecture and then describe the problem formulation for the QoS routing schema.

In SDN, the controller maintains a global knowledge of the network which provides the possibility to have an efficient network management schema. Thus, our architecture leverages the advantages of SDN in an IoUT environment. Figure 2 illustrates our proposed architecture. According to this figure, the SDN controller located above the sea surface manages the underwater things that have been deployed in the underwater environment. An underwater thing can be a sensor, an AUV or any other smart equipment. These things are able to communicate with each other in order to exchange the information collected from their surrounding environment. Furthermore, these underwater objects are capable pf forwarding packets traveling from a source toward the sink. To do this, each thing has a flow table similar to [17] and it can be programmed to execute the operations specified by the remote controller. In fact, the underwater objects can act as a forwarder (i.e. a switch). This means that when a node receives a packet, it forwards the packet according to the forwarding rules in its flow table. The flow table is filled or emptied according to the instructions sent by the controller. In this architecture, the SDN controller is implemented on the sink node and in addition to collecting data from underwater objects, it is responsible for controlling network devices and installing paths. In fact, to install the paths on the relevant nodes, the controller sends forwarding rules to the nodes along a path and asks them to add the rules to their flow table. After this step, the nodes can forward the receiving packets regarding the rules. In our proposed architecture, the controller is accessible and configurable from anywhere in the world via the Internet connection. In addition to the current capabilities, a network administrator can add more functionalities such as load balancing, security, energy saving mechanisms, etc.

The proposed architecture is beneficial because the controller can collect network statistics and offer efficient routes to underwater things based on the current condition of underwater channel. This achievement is due to the use of a centralized component which alleviates the problems of distributed schemes. As the matter of fact, it is difficult to control

**Table 1** Comparison between related work

| Paper | SDN based | IoT based | Energy | Delay | Reliability | Mobility |
|---|---|---|---|---|---|---|
| Berlian et al. [20] | X | ✓ | X | X | X | X |
| Qin et al. [32] | X | ✓ | ✓ | X | X | ✓ |
| Zhou et al. [21] | X | ✓ | ✓ | X | X | X |
| Torres et al. [30] | ✓ | X | X | ✓ | ✓ | X |
| Nayyar et al. [22] | X | ✓ | X | X | X | X |
| Urunov et al. [23] | X | ✓ | ✓ | X | X | X |
| Akyildiz et al. [17] | ✓ | X | ✓ | ✓ | ✓ | ✓ |
| Kao et al. [24] | X | ✓ | X | X | ✓ | X |
| Francisco et al. [25] | X | ✓ | ✓ | X | ✓ | X |
| Luo et al. [34] | ✓ | X | ✓ | ✓ | ✓ | X |
| Xu and Liu [26] | X | ✓ | ✓ | X | ✓ | X |
| wang et al. [31] | ✓ | X | X | ✓ | ✓ | X |
| Fan et al. [35] | ✓ | X | X | X | ✓ | ✓ |
| Lin et al. [33] | ✓ | X | X | ✓ | ✓ | ✓ |
| Qin et al. [32] | X | ✓ | ✓ | X | X | ✓ |
| Alostad [36] | X | ✓ | ✓ | X | ✓ | X |
| Proposed method | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

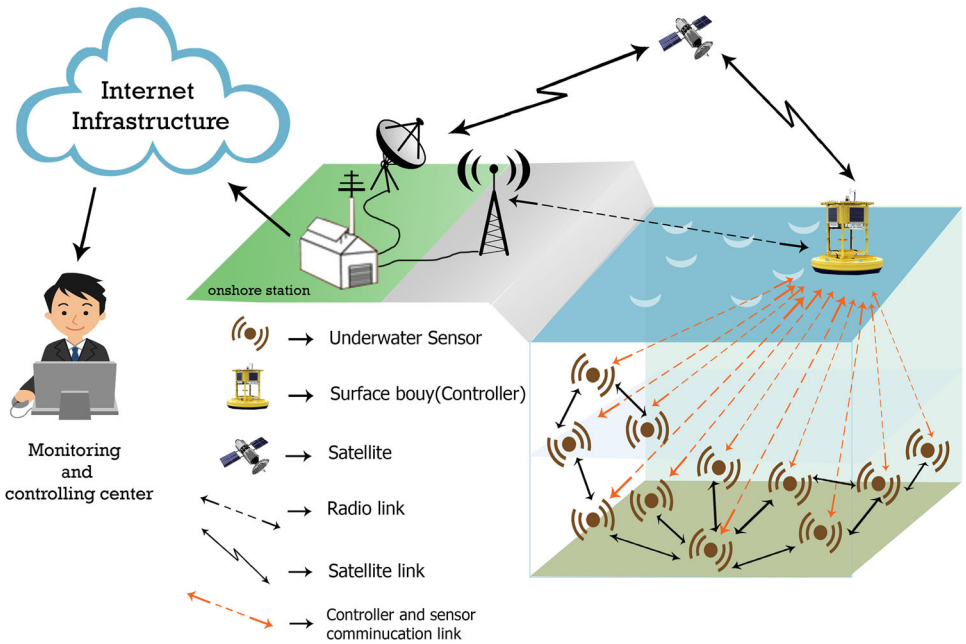



**Fig. 2** Proposed SDN-based IoUT architecture

underwater things using a distributed mechanism especially when the scale of the network expands.

### 3.1 Problem formulation

The role of the controller is to calculate and announce a reliable route in order to increase the packet delivery ratio and to decrease the end-to-end delay to achieve a better quality of service. Regarding the critical role of the controller, it is worth to mention that, it can be possible to add additional backup controllers to eliminate single point of failure problem. In the following, we describe how the controller calculates a reliable path between two pair of underwater nodes.

We assume that each node knows its coordinates and periodically announces the coordinate information to the controller and there are also no two collocated nodes in the environment. It should be noted that the coordinate of each thing can be calculated using various localization methods [37]. In our proposed schema, we assume that the movement of nodes is little and negligible. After obtaining the coordinate data of all the underwater things, the controller can calculate the distance between a pair of nodes using Eq. (1):

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \quad (1)$$

where $d$ is the Euclidean distance between the nodes. Afterward, the controller must calculate the successful delivery ratio for a packet. To do this, it uses Eq. (2) as follows [11,24]:

$$P^m_{successful}(\gamma) = \left[1 - BER(\gamma)\right]^m = \left[P^1_{successful}(\gamma)\right]^m = \left\{\frac{1}{2} + \frac{1}{2}\sqrt{\frac{10^{\gamma/10}}{1 + 10^{\gamma/10}}}\right\}^m \quad (2)$$

where $m$ is the packet size, is the Signal-to-Noise Ratio (SNR) and $BER()$ is the Bit Error Rate of the underwater channel, respectively. The SNR of an underwater channel is obtained from Eq. (3) [11,24].

$$\gamma = 10[\log(P) - \log(4\pi.r^2) - \log(0.67 \times 10^{-18})] - 20\log(d) - \left[\left(\frac{0.11 f^2}{1 + f^2} + \frac{44 f^2}{4100 + f^2} + 2.75 \times 10^{-4} f^2 + 0.003\right) \times d \times 10^{-3}\right] - 50 + 18\log(f) \quad (3)$$

where $P$ is the transmission power, $d$ is the distance between the two nodes which is calculated using Eq. (1), $f$ is the frequency and $r$ is the transmission rage. All parameters in Eq. (3) are channel specific and can be manually configured by the administrator in the controller. We assume that all underwater things have the same transmission power, the same transmission range and use the same frequency band for communication.

Till now, the controller knows the distance between adjacent nodes and the reliability of all links between the nodes. To compute reliable paths between the nodes, the controller can consider a cost for each link by combining the delay and reliability of the link as follows [11, 24]:

$$\omega_{ab} = \alpha.d_{ab} + (1 - \alpha).p_{ab} \quad (4)$$

where is weighting factor,$\mathbf{d_{ab}}$ is the distance between two nodes (calculated in Eq. 1) and $\mathbf{p_{ab}}$ is the probability of packet successful delivery ratio (calculated in Eq. 2). It is worth noting that, before using Eq. (4), the controller normalizes $\mathbf{p_{ab}}$ and $\mathbf{d_{ab}}$, and then use them in the formula. After calculating the cost for each link, the controller applies the Dijkstra algorithm to find the minimum cost path for a pair of nodes. The minimum cost path is the path with the highest reliability and shortest delay. At the final step, the controller installs the calculated path on the relevant nodes. After these steps, upon receiving a packet, each node will be able to send it to the next determined hop.

**Algorithm 1:** Pseudo-code for path computation in the controller.

**Data**: $\mathcal{N}odes$: list of sensor nodes, $Coordinates$: list of nodes' coordinates, $\mathcal{W}$: Matrix of Cost between two nodes, $\mathcal{R}$: Transmission Range of each node, $\mathcal{P}m$: Path from node m toward the sink node, $\mathcal{D}ijkstra(W, m)$:Shortest path between m and sink node regarding $\mathcal{W}$

1 $\mathcal{W} \leftarrow 0$ ; /* Initialize Cost matrix */
2 **for** $m \in \mathcal{N}odes$ **do**
3  **for** $n \in \mathcal{N}odes$ **do**
4  **if** $(m! = n)$ **then**
5  $d_{mn} \leftarrow$ distance(m,n) ; /* Compute distance using Eq.1 */
6  **if** $(d_{mn} > \mathcal{R})$ **then**
7  $\mathcal{W}[m][n] \leftarrow 0$
8  **else**
 /* Calculate packet delivery ratio between m and n using Eq.2 */
9  $p_{mn} \leftarrow$ PDR(m,n)
10  $\mathcal{W}[m][n] \leftarrow \alpha.d_{mn} + (1 - \alpha).p_{mn}$
11  **end**
12  **else**
13  $\mathcal{W}[m][n] \leftarrow 0$
14  **end**
15  **end**
16 **end**
17 **for** $m \in \mathcal{N}odes$ **do**
18  $\mathcal{P}m \leftarrow \mathcal{D}ijkstra(W, m)$
19  install($\mathcal{P}m$) /* Install $\mathcal{P}m$ on all relevant nodes along the path */
20 **end**

In order to eliminate the complexity of network management, we assume that the controller operates proactively rather than reactively. In this way, before any communication between the nodes, the controller calculates and installs the paths between the nodes. Since in an underwater environment most nodes have mobility due to water currents and the environment is also very variable, we consider that the controller periodically performs the above process to announce new paths with respect to the current condition of the underwater channel. The Algorithm 1 shows the pseudo-code for calculating the path between underwater things and the sink node. In Algorithm 1, from line 2 to 16, the controller fetches the link's length and reliability and then calculates the cost of each link using Eq. (4). *if* condition in line 6, prevents the cost calculation for the links whose length is greater than the sensor transmission range. From line 17 to 20, the controller calls *Dijkstra(W,m)* function in order to calculate the minimum cost path between the network nodes.

## 4 Performance evaluation

In this section, a comprehensive simulation study is performed in MATLAB [38]. The results were analyzed to assess the performance of the proposed method and to compare with two famous underwater routing schemes, namely Depth-Based Routing (DBR) [39] and Broadcast Routing. DBR is a well-known routing protocol in underwater networks and most of the literature have been used DBR as a benchmark to evaluate their works. In DBR, each node upon receiving a packet, first examines the depth field in the header of the received packet which has been set by the previous node. If this value is greater than the depth value of current node, the packet is forwarded to upper nodes. Otherwise, the packet will be dropped. By this way, the packet is routed from its originating node to the sink node on the water surface. Broadcast routing is also another popular benchmark in which each node simply broadcasts the received packets without any consideration. In addition to DBR and Broadcast routing, we have implemented a Hop Based Routing (HBR) protocol that calculates the route from source to destination based on the number of hops between them. The use of hop-count metric for routing is very popular in computer networks, and most of the known routing protocols use this concept.

To make a fair comparison, the proposed method and benchmarks are evaluated in different scenarios. All experiments are executed on a computer with a CPU Intel Core i7-4700MQ-2.4 GHz and 6 GB of memory. For all simulations, the sink node collects transmitted packets from underwater things and also operates as the controller. In our simulations, the sink is located in the center of the topology on the water surface while other underwater things are randomly deployed in a 3-D underwater $500m \times 500m \times 500m$ area. Each underwater thing announces it's coordinate every 20 s to the sink. Accordingly, the sink is configured to execute the computation every 20 s and then announces the calculated paths to the underwater things. We assume the configuration parameters similar to LinkQuest UWM1000 [40] acoustic modem for all underwater nodes. All simulation parameters are shown in Table 2.

We assume the following metrics for the performance evaluation.

- *Average End-to-End Delay:* it is the average elapsed time that a packet travels from an underwater thing to the sink node.
- *Packet Loss Ratio:* it is the ratio between the number of packets not delivered to the sink node and all the packets originated from the underwater nodes.
- *Energy Consumption:* it denotes the total energy consumed by underwater things for sending and receiving.

**Table 2** Simulation Parameters

| Parameters | Value |
|---|---|
| Number of underwater things | 500 |
| Simulation time | 3600 s |
| Frequency | 26.7 KHz |
| Transmit mode power consumption | 2 W |
| Receive mode power consumption | 0.75 W |
| Bandwidth | 17.8 kbps |
| Communication range | 350 m |
| Packet size | 100 Byte |

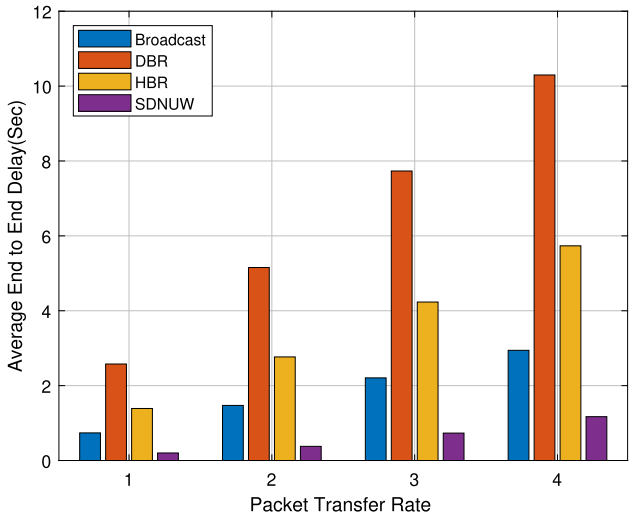
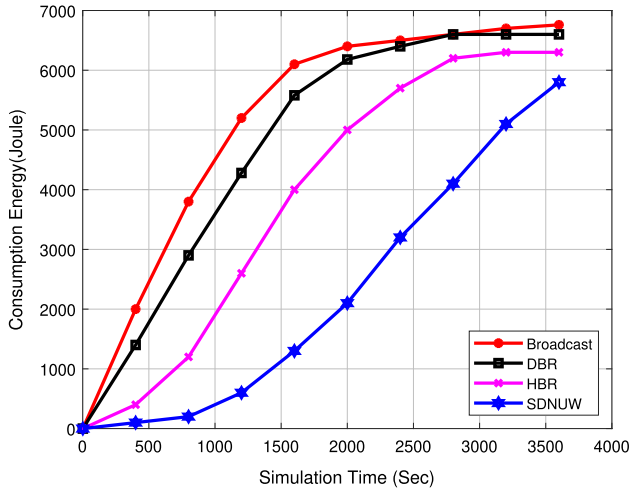


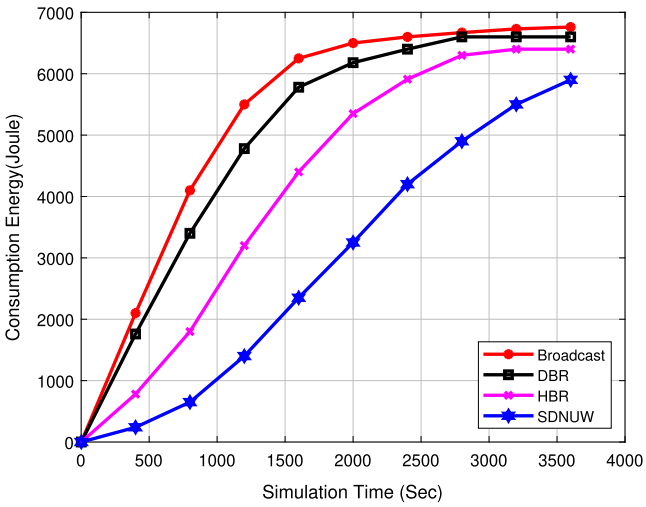

**Fig. 3** Average end-to-end delay versus traffic rate

- *First Node Death Time:* it denotes the time the first underwater node runs out of energy.
- *Number of Alive Nodes:* it indicates the number of underwater things for which the battery is not exhausted at the end of the simulation.

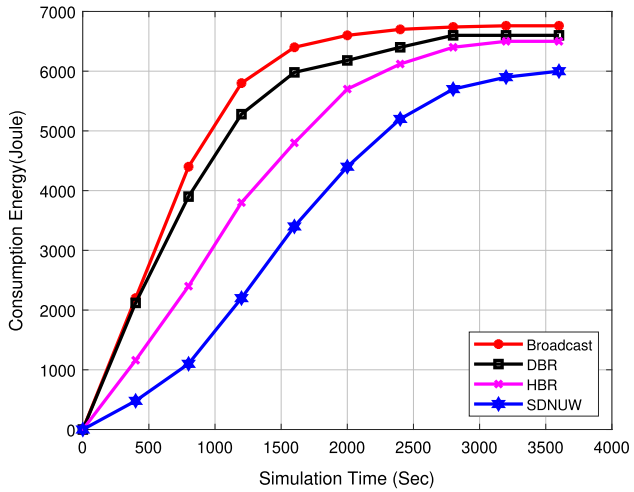
Figure 3 shows that the proposed method achieves reasonable performance in terms of average end-to-end delay for different traffic rates and outperforms other three methods. The reason is that, the cost of each link is calculated according to it's delay. On the other hand, in DBR, each node postpones packet forwarding for a specific time leading to an increase in the average end-to-end delay (cf. Fig. 3). Figure 3 also shows that Broadcast and HBR routing obtains longer average end-to-end delay than the proposed method for all traffic rates. The reason is that, in Broadcast routing each node immediately forwards a received packet to all it's neighbors. As a result, the network will be congested and it takes longer in MAC layer for a device to capture the channel in order to forward the received packets. For HBR, the average end-to-end delay is relatively high, as it does not consider the distance



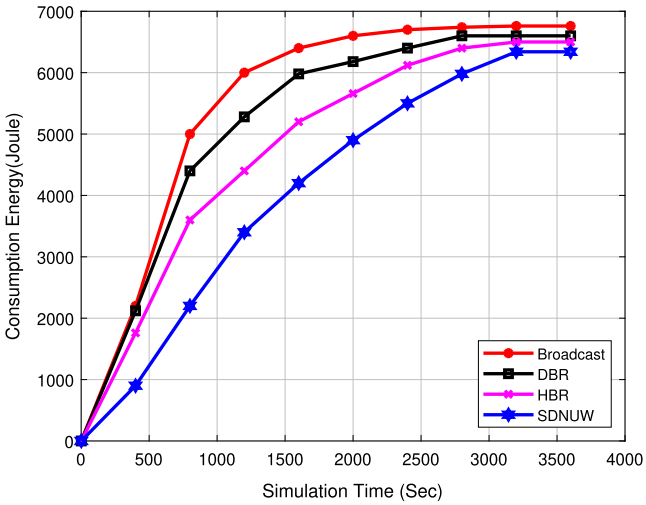

**(a)** Traffic rate (1 packet per second)



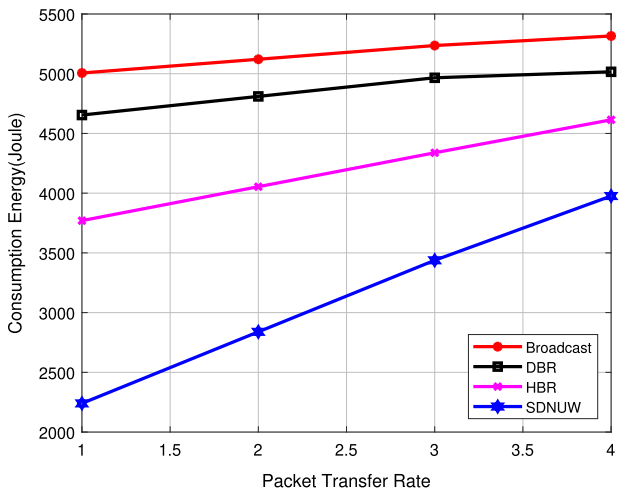

**(b)** Traffic rate (2 packets per second)




**(c)** Traffic rate (3 packets per second)




**(d)** Traffic rate (4 packets per second)




**(e)** Different traffic rates in a topology with 500 nodes

**Fig. 4** Energy consumption versus traffic rate

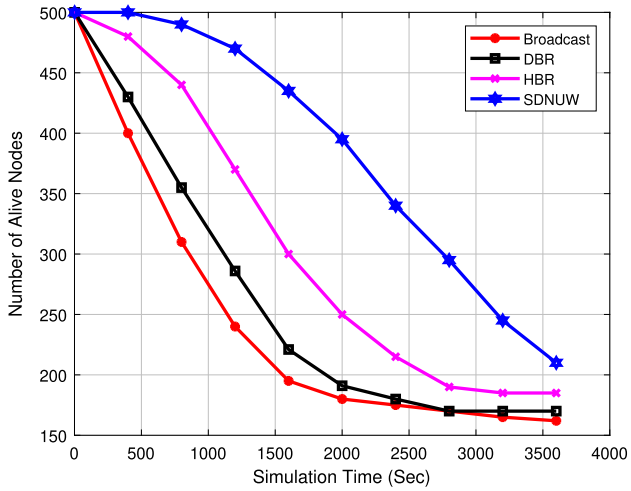
or delay between nodes and only chooses the shortest path in terms of the number of hops.

As mentioned in Sect. 1, energy efficiency is one of the main important issues for underwater things. Since in the proposed method, each thing notifies the controller only of it's coordinate and only executes the commands received from the controller, it can save energy as shown in Fig. 4. This figure shows that not carrying out additional operations by underwater things when using the proposed methods leads to greater energy efficiency compared to other methods for different node densities and traffic rates. In Broadcast routing, each thing inefficiently broadcasts the received packet and thus consumes more energy. However, in DBR, each node takes some consideration in forwarding the packets and therefore consumes less energy than pure Broadcast routing. The reason why HBR consumes less energy than DBR and Broadcast routing, is that it minimizes the number of nodes along the path, and as the result, less energy is consumed per node. As expected, in Fig. 4e, energy consumption increases when traffic rate increases.

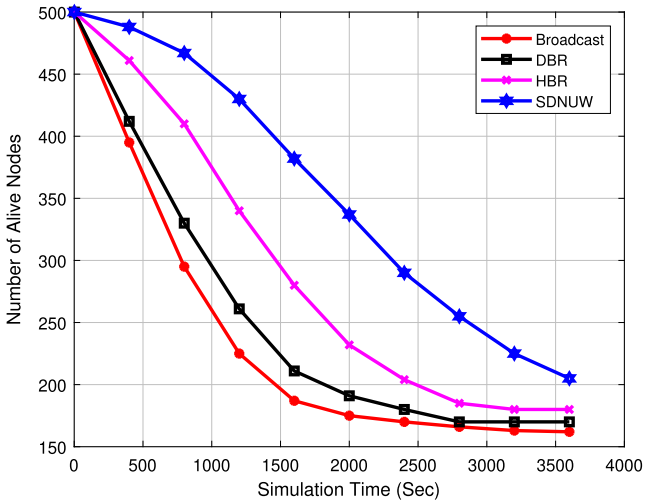
Figure 5 shows the number of alive nodes during the simulation. As it is observed from the figure, the proposed method outperforms the other methods for different number of nodes and traffic rates. The reason is that in the proposed method, each node only follows the commands received from the controller and takes no further action. This policy leads to a drop in energy consumption which causes more aliveness. As we discussed in Sect. 3, the network controller determines the next-hop neighbor for the nodes. Therefore, when receiving a packet, a node does not broadcast it, instead, it puts the address of the next hop node and forwards it according to the forwarding rule in its flow table. This behavior limits the forwarding nodes, and as the result, it leads to less energy consumption. Unlike our proposed method, in Broadcast routing, each node blindly broadcasts the received packets to all its neighbors. Therefore, the battery of nodes is depleted sooner. For DBR, because each node only broadcasts the received packet if it comes from a lower depth, the number of alive nodes is more than pure Broadcast routing, and also it achieves longer lifetime than Broadcast routing in all scenarios. Since HBR does not broadcast the packets and only forwards them to an specific next hop, it outperforms DBR and Broadcast routing in terms of number of alive nodes. In Fig. 5a–d, after 1500–200 s, the number of alive nodes is almost constant without considerable decrease or increment. The reason is because of after the dead of traffic producer nodes, the alive nodes do not perform forwarding and hence the number of them does not change. As Fig. 5e depicts, the number of alive nodes decreases as the traffic rate increases. It is obvious that in a high traffic rate, each node consumes more energy the source node generates more traffic and more nodes contribute in forwarding the packets, and lead to more energy consumption.

The death of a node in the network can divide it into several parts or cause difficulties such as a longer delay or a higher packet loss rate. As shown in Fig. 6, in any case, for the proposed method, the death time of the first node is longer than the other three methods. The reason behind this is that the proposed method does not impose additional operations taking more energy consumption on the underwater things. This policy extends the life of underwater objects. Unlike the proposed method, broadcast routing and DBR have no mechanism to decrease energy consumption, and they only broadcast packets without setting the destination address. This behavior leads to a situation where all neighbors around a specific node broadcast packets to their neighbors. As the result, more nodes involve packet routing and the energy consumption increases considerably. The probability of involving in routing the packets for the nodes located in the middle of the network is high, and the energy consumption for these nodes is higher than the other nodes. Therefore, the death time of these nodes is earlier than the others in Broadcast routing and DBR. As a result, the death time of the first node in these two methods in all cases is earlier than HBR and the proposed method. Figure 6, also confirms that, since HBR involves a low number of nodes in routing, it performs better than DBR and Broadcast routing.

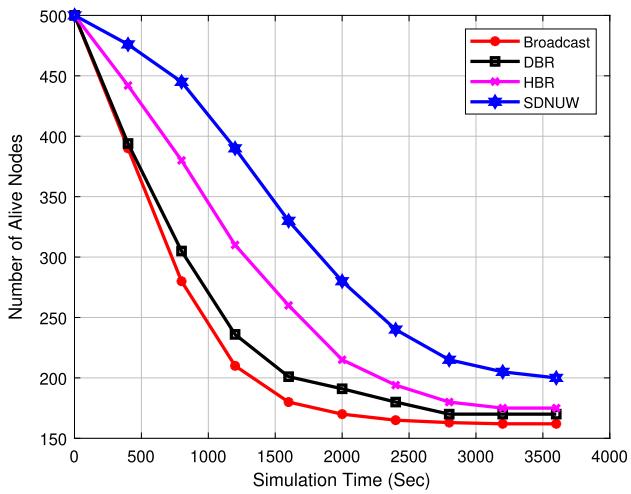
As we mentioned in Sect. 3, one of the main goals of our proposed method is to increase the reliability of communication in IoUT. To achieve this goal, we used a mathematical model to estimate the probability of packet delivery ratio and then calculate the cost of the link. Figure 7 proves that this technique can considerably reduce the packet loss ratio for different traffic rates, because it estimates high cost for unreliable links. As the result, in route calculation step, the routing schema does not include unreliable links for paths, and attempts to calculate paths with low cost. Therefore, the calculated path comprises the links with lower cost (i.e. higher reliability). Moreover, as we discussed for Fig. 6, the earlier death time for an underwater thing may cause unreliability for a network in terms of path failure. In fact, if a node located along a path between the source node and the sink node dies, the path will be broken, and as the result, all packets flowing along the path will be dropped. For this reason, a longer death time of the first node also increases the reliability of the proposed method and as a consequence it achieves less packet loss ratio. Figure 7 also illustrates that because in Broadcast routing, there is no mechanism to prevent collisions, the packet loss ratio is the highest. DBR has a lower packet loss ratio than Broadcast routing, because in DBR a received packet is forwarded only when it comes form a lower depth. Furthermore, according to Fig. 6, the death time of the first node in Broadcast routing and DBR is short, and leads to more unreliability. From Figs. 7 and 6, it can be concluded that HBR performs better than Broadcast and DBR routing, because unlike these two protocols, it



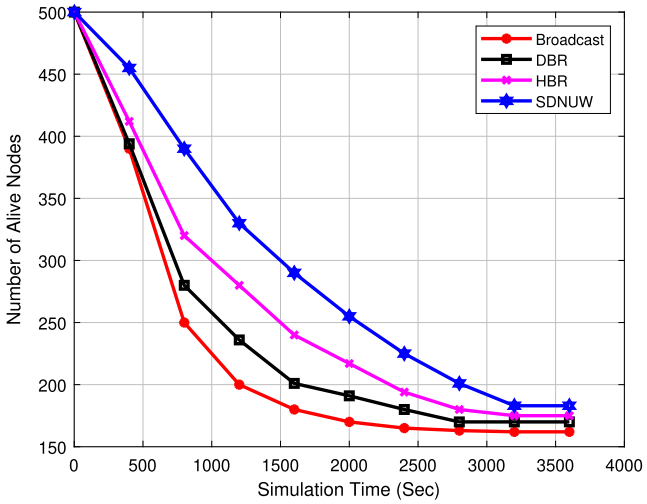

**(a)** Traffic rate (1 packet per second)



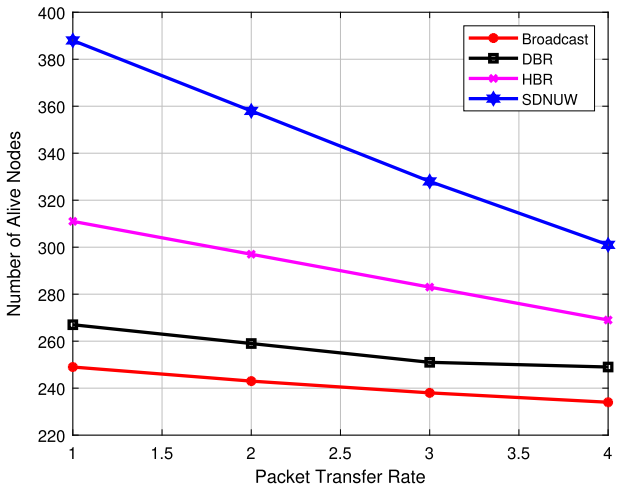

**(b)** Traffic rate (2 packets per second)




**(c)** Traffic rate (3 packets per second)

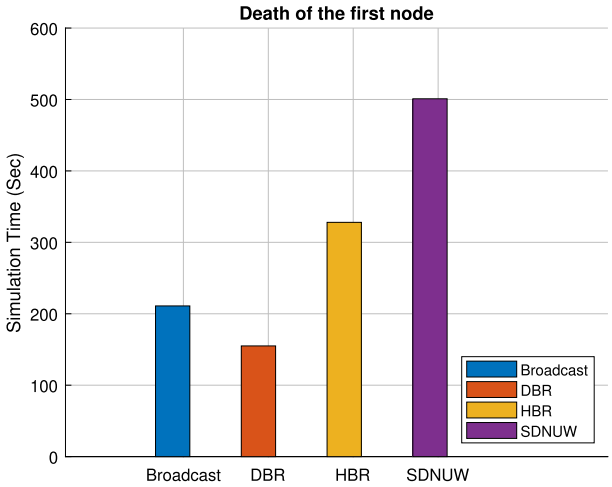



**(d)** Traffic rate (4 packets per second)

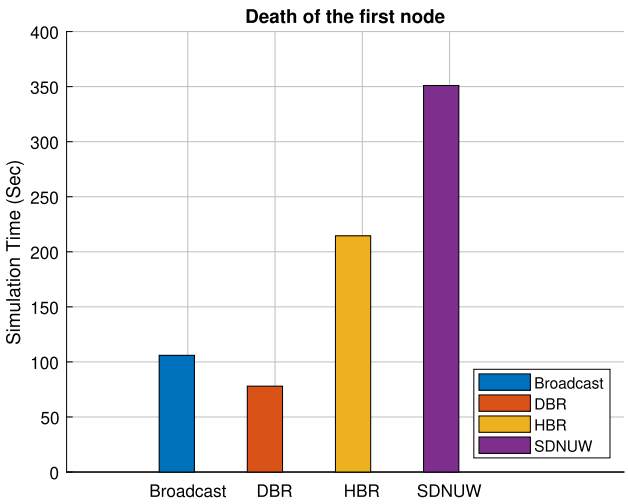



**(e)** Different traffic rates in a topology with 500 nodes

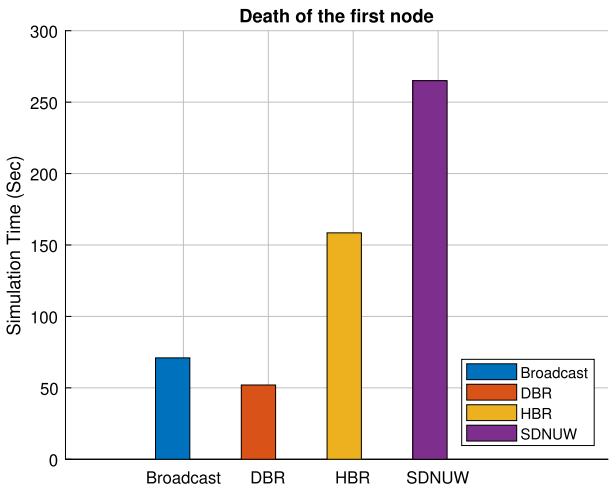
**Fig. 5** Number of alive nodes versus traffic rate



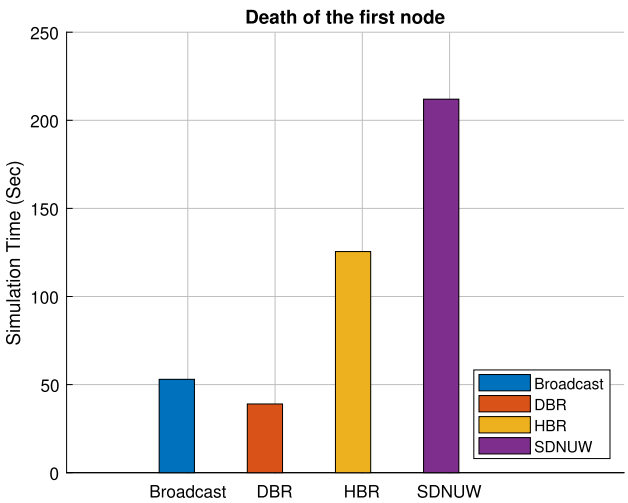

**(a)** Traffic rate (1 packet per second)



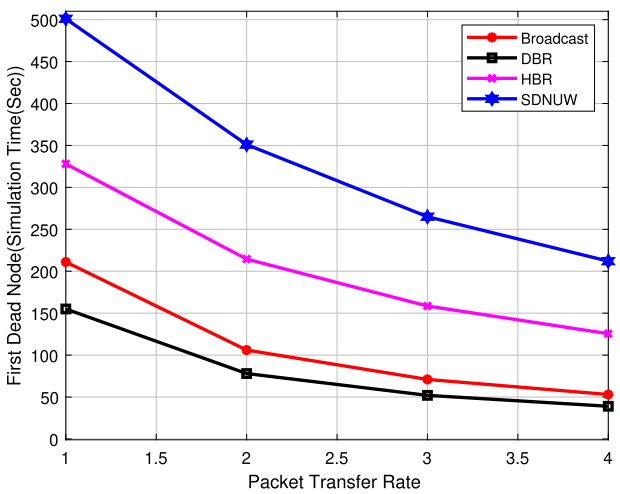

**(b)** Traffic rate (2 packets per second)




**(c)** Traffic rate (3 packets per second)




**(d)** Traffic rate (4 packets per second)




**(e)** Different traffic rates in a topology with 500 nodes

**Fig. 6** First node death time versus traffic rate

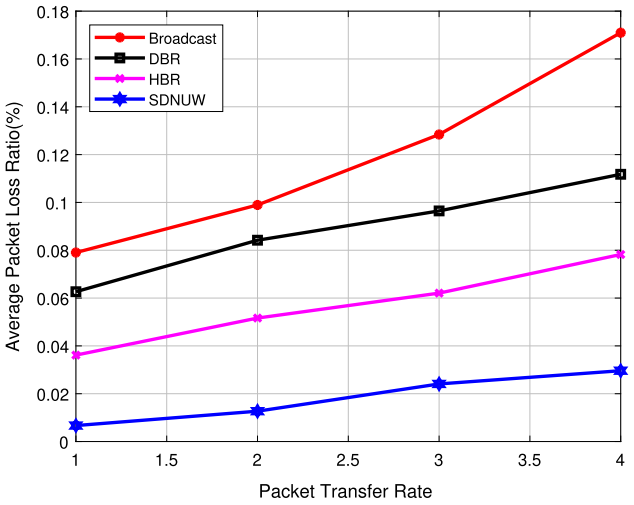



**Fig. 7** Packet loss ratio versus traffic rate in a topology with 500 nodes

does not broadcast packets and reduce the probability of collision, and therefore, leads to lower packet loss ratio. But, in comparison with the proposed method, since it does not take unreliable links into account, its packet loss ratio remains high.

## 5 Conclusions

In this paper, a new SDN-based architecture for IoUT with the goal of leveraging the benefits of SDN to provide QoS was proposed. Furthermore, a new routing schema was introduced to make use of SDN capabilities to find an appropriate path between underwater things. The proposed routing method first collects the coordinate of each node, then estimates the reliability and delay of each link. Afterward, it calculates the appropriate paths and finally installs the paths on the underwater things. Simulation results showed that the application of SDN architecture to the IoUT considerably facilitates the implementation of a centralized routing schema allowing optimal decisions to be made. Simulation results also proved that the deployment of an SDN-based architecture together with condition-aware routing policies can help network providers to manage their IoUT in a more flexible and efficient manner which leads to a low average end-to-end delay, longer network lifetime and less packet loss ratio.

## References

1. Luo, H., Wu, K., Ruby, R., Liang, Y., Guo, Z., & Ni, L. M. (2018). Software-defined architectures and technologies for underwater wireless sensor networks: a survey. *IEEE Communications Surveys & Tutorials*, *20*(4), 2855–2888.
2. Akyildiz, I. F., Pompili, D., & Melodia, T. (2005). Underwater acoustic sensor networks: Research challenges. *Ad hoc Networks*, *3*(3), 257–279.
3. Mohammadi, R., & Javidan, R. (2016). A comparative study on mac protocols of an underwater surveillance system from qos perspective. *Journal of Telecommunication Electronic and Computer Engineering (JTEC)*, *8*(1), 45–52.
4. Manjula, R., & Manvi, S. S. (2011). Issues in underwater acoustic sensor networks. *International Journal of Computer and Electrical Engineering*, *3*(1), 101.
5. Stojanovic, M. (1999). Underwater acoustic communication. *Wiley Encyclopedia of Electrical and Electronics Engineering*, *19*, 1–12.
6. Etter, P. C. (2014). *Underwater acoustic modeling: principles, techniques and applications*. Baco Raton: CRC Press.
7. Keshtgary, M., Mohammadi, R., Mahmoudi, M., & Mansouri, M. R. (2012). Energy consumption estimation in cluster based underwater wireless sensor networks using m/m/1 queuing model. *International Journal of Computer Applications*, *43*(24), 6–10.
8. Stojanovic, M. (2003) "Acoustic (underwater) communications," *Wiley Encyclopedia of Telecommunications*.
9. Awan, K. M., Shah, P. A., Iqbal, K., Gillani, S., Ahmad, W., & Nam, Y. (2019). Underwater wireless sensor networks: A review of recent issues and challenges. *Wireless Communications and Mobile Computing*.
10. Stojanovic, M., & Preisig, J. (2009). Underwater acoustic communication channels: Propagation models and statistical characterization. *IEEE Communications Magazine*, *47*(1), 84–89.
11. Kao, C.-C., Lin, Y.-S., Wu, G.-D., & Huang, C.-J. (2017). A comprehensive study on the internet of underwater things: applications, challenges, and channel models. *Sensors*, *17*(7), 1477.
12. Domingo, M. C. (2012). An overview of the internet of underwater things. *Journal of Network and Computer Applications*, *35*(6), 1879–1890.
13. Shin, M.-K., Nam, K.-H., & Kim, H.-J. (2012) Software-defined networking (sdn): A reference architecture and open apis. In: *2012 International Conference on ICT Convergence (ICTC)*, pp. 360–361, IEEE.
14. Keshari, S.K., Kansal, V., & Kumar, S. (2020) A systematic review of quality of services (qos) in software defined networking (sdn).*Wireless Personal Communications*, pp. 1–22.
15. Kobo, H. I., Abu-Mahfouz, A. M., & Hancke, G. P. (2017). A survey on software-defined wireless sensor networks: Challenges and design requirements. *IEEE Access*, *5*, 1872–1899.
16. Demirors, E.,Shi, J.,Duong, A., Dave, N., Guida, R., Herrera, B.,Pop, F., Chen, G., Casella, C. & Tadayon, S. et al. (2018) The seanet project: Toward a programmable internet of underwater things. In *2018 Fourth Underwater Communications and Networking Conference (UComms)*, pp. 1–5, IEEE
17. Akyildiz, I. F., Wang, P., & Lin, S.-C. (2016). Softwater: Software-defined networking for next-generation underwater communication systems. *Ad Hoc Networks*, *46*, 1–11.
18. Sharma, C. (2016) Correcting the iot history
19. Meddeb, A. (2016). Internet of things standards: who stands out from the crowd? *IEEE Communications Magazine*, *54*(7), 40–47.
20. Berlian, M.H., Sahputra, T.E.R., Ardi, B.J.W., Dzatmika, L.W., Besari, A.R.A., Sudibyo, R.W., & Sukaridhoto, S. (2016) "Design and implementation of smart environment monitoring and analytics in real-time system framework based on internet of underwater things and big data. In: *2016 International Electronics Symposium (IES)*, pp. 403–408, IEEE.
21. Zhou, Z., Yao, B., Xing, R., Shu, L., & Bu, S. (2015). E-carp: An energy efficient routing protocol for uwsns in the internet of underwater things. *IEEE Sensors Journal*, *16*(11), 4072–4082.
22. Nayyar, A., Ba, C.H., Duc, N.P.C., & Binh, H.D. (2018) Smartiout 1.0: A smart aquatic monitoring network based on internet of

underwater things (iout). In *International Conference on Industrial Networks and Intelligent Systems*, pp. 191–207. Springer, Berlin.
23. Urunov, K., Shin, S.-Y., Namgung, J.-I., & Park, S.-H. (2018) High-level architectural design of management system for the internet of underwater things. In: *2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 326–331, IEEE.
24. Kao, C.-C., Lin, Y.-S.,Wu, G.-D. , & Huang, C.-J. (2017) A study of applications, challenges, and channel models on the internet of underwater things. In: *2017 International Conference on Applied System Innovation (ICASI)*, pp. 1375–1378, IEEE.
25. Lima, F. H., Vieira, L. F., Vieira, M. A., Vieira, A. B., & Nacif, J. A. M. (2019). Water ping: Icmp for the internet of underwater things. *Computer Networks*, *152*, 54–63.
26. Xu, M., & Liu, L. (2016) Sender-receiver role-based energy-aware scheduling for internet of underwater things. *IEEE Transactions on Emerging Topics in Computing*
27. Luo, Y., Pu, L., Zuba, M., Peng, Z., & Cui, J.-H. (2014). Challenges and opportunities of underwater cognitive acoustic networks. *IEEE Transactions on Emerging Topics in Computing*, *2*(2), 198–211.
28. Jones, E. (2007) The application of software radio techniques to underwater acoustic communications. In *OCEANS 2007-Europe*, pp. 1–6, IEEE.
29. Torres, D., Friedman, J., Schmid, T., & Srivastava, M.B. (2009) Software-defined underwater acoustic networking platform. In: *Proceedings of the fourth ACM international workshop on underwater networks*, p. 7, ACM.
30. Torres, D., Friedman, J., Schmid, T., Srivastava, M. B., Noh, Y., & Gerla, M. (2015). Software-defined underwater acoustic networking platform and its applications. *Ad Hoc Networks*, *34*, 252–264.
31. Wang, J., Ma, L., & Chen, W. (2017). Design of underwater acoustic sensor communication systems based on software-defined networks in big data. *International Journal of Distributed Sensor Networks*, *13*(7), 1550147717719672.
32. Qin, C., Du, J., Wang, J., & Ren, Y. (2020). A hierarchical information acquisition system for auv assisted internet of underwater things. *IEEE Access*, *8*, 176089–176100.
33. Lin, C., Han, G., Guizani, M., Bi, Y., & Du, J. (2019). A scheme for delay-sensitive spatiotemporal routing in sdn-enabled underwater acoustic sensor networks. *IEEE Transactions on Vehicular Technology*, *68*(9), 9280–9292.
34. Luo, H., Liu, C., & Liang, Y. (2019) A sdn-based testbed for underwater sensor networks
35. Fan, R., Wei, L., Du, P., McGoldrick, C., Gerla, M. (2016) A sdn-controlled underwater mac and routing testbed. In: *MILCOM 2016-2016 IEEE Military Communications Conference*, pp. 1071–1076, IEEE.
36. Alostad, J. M. (2020). Reliability in iout enabled underwater sensor networks using dynamic adaptive routing protocol. *International Journal of Internet Manufacturing and Services*, *7*(1-2), 115–129.
37. Tuna, G., & Gungor, V. C. (2017). A survey on deployment techniques, localization algorithms, and research challenges for underwater acoustic sensor networks. *International Journal of Communication Systems*, *30*(17), e3350.
38. https://www.mathworks.com/products/matlab.html, "Matlab,"
39. Yan, H., Shi, Z.J., & Cui, J.-H. (2008) Dbr: depth-based routing for underwater sensor networks. In:*International conference on research in networking*, pp. 72–86, Springer, Berlin.
40. https://www.linkquest.com, "Linkquest modem,"



**Reza Mohammadi** is an Assistant Professor in Computer Engineering at Bu-Ali Sina University since 2018. He received his MSc and Ph.D. degrees in Computer Networking from Shiraz University of Technology in 2013 and 2017, respectively. His PhD thesis was concerned with traffic engineering in Software Defined Networks (SDN). He has several publications in international conferences and journals regarding Underwater Wireless Sensor Networks (UWSNs) and Software Defined Networks (SDNs). His major fields of interest are SDN, heuristic algorithms, SDN security, Underwater Wireless Sensor Networks, Ad hoc Networks, Underground Wireless Sensor Networks, Internet of Things (IoT) and IoUT.

**Amin Nazari** received BSc degree in Computer Software Engineering from Islamic Azad University, Hamedan, in 2009. He received his MSc degree in Computer Software Engineering from Arak University, Arak, in 2015. He is currently working toward a research assistant at the Department of Computer Engineering in Bu-Ali Sina University, Hamedan. His research interests include wireless sensor networks, the Internet of Things and IoT-fog networks.

**Mohammad Nassiri** is currently an associate professor at Bu-Ali Sina University. After having graduated from Sharif University of Technology (Iran), he obtained his Ph.D. in computer science from Grenoble INP (France) in 2008. He was a visiting researcher at the Universite Grenoble Alpes (France) during summer 2019. His research interests mainly concern improving the performance of various wireless technologies, namely wireless LANs, wireless sensor networks, underwater networks, and interconnecting technologies for the Internet of Things.

**Mauro Conti** is Full Professor at the University of Padua, Italy. He is also affiliated with TU Delft and University of Washington, Seattle. He obtained his Ph.D. from Sapienza University of Rome, Italy, in 2009. After his Ph.D., he was a Post-Doc Researcher at Vrije Universiteit Amsterdam, The Netherlands. In 2011 he joined as Assistant Professor the University of Padua, where he became Associate Professor in 2015, and Full Professor in 2018. He has been Visiting Researcher at GMU, UCLA, UCI, TU Darmstadt, UF, and FIU. He has been awarded with a Marie Curie Fellowship (2012) by the European Commission, and with a Fellowship by the German DAAD (2013). His research is also funded by companies, including Cisco, Intel, and Huawei. His main research interest is in the area of Security and Privacy. In this area, he published more than 400 papers in topmost international peer-reviewed journals and conferences. He is Area Editor-in-Chief for IEEE Communications Surveys Tutorials, and has been Associate Editor for several journals, including IEEE Communications Surveys & Tutorials, IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, and IEEE Transactions on Network and Service Management. He was Program Chair for TRUST 2015, ICISS 2016, WiSec 2017, ACNS 2020, and General Chair for SecureComm 2012, SACMAT 2013, CANS 2021, and ACNS 2022. He is Senior Member of the IEEE and ACM. He is a member of the Blockchain Expert Panel of the Italian Government. He is Fellow of the Young Academy of Europe.

# Terms and Conditions

Springer Nature journal content, brought to you courtesy of Springer Nature Customer Service Center GmbH (“Springer Nature”).
Springer Nature supports a reasonable amount of sharing of research papers by authors, subscribers and authorised users (“Users”), for small-scale personal, non-commercial use provided that all copyright, trade and service marks and other proprietary notices are maintained. By accessing, sharing, receiving or otherwise using the Springer Nature journal content you agree to these terms of use (“Terms”). For these purposes, Springer Nature considers academic use (by researchers and students) to be non-commercial.
These Terms are supplementary and will apply in addition to any applicable website terms and conditions, a relevant site licence or a personal subscription. These Terms will prevail over any conflict or ambiguity with regards to the relevant terms, a site licence or a personal subscription (to the extent of the conflict or ambiguity only). For Creative Commons-licensed articles, the terms of the Creative Commons license used will apply.
We collect and use personal data to provide access to the Springer Nature journal content. We may also use these personal data internally within ResearchGate and Springer Nature and as agreed share it, in an anonymised way, for purposes of tracking, analysis and reporting. We will not otherwise disclose your personal data outside the ResearchGate or the Springer Nature group of companies unless we have your permission as detailed in the Privacy Policy.
While Users may use the Springer Nature journal content for small scale, personal non-commercial use, it is important to note that Users may not:

1. use such content for the purpose of providing other users with access on a regular or large scale basis or as a means to circumvent access control;
2. use such content where to do so would be considered a criminal or statutory offence in any jurisdiction, or gives rise to civil liability, or is otherwise unlawful;
3. falsely or misleadingly imply or suggest endorsement, approval , sponsorship, or association unless explicitly agreed to by Springer Nature in writing;
4. use bots or other automated methods to access the content or redirect messages
5. override any security feature or exclusionary protocol; or
6. share the content in order to create substitute for Springer Nature products or services or a systematic database of Springer Nature journal content.

In line with the restriction against commercial use, Springer Nature does not permit the creation of a product or service that creates revenue, royalties, rent or income from our content or its inclusion as part of a paid for service or for other commercial gain. Springer Nature journal content cannot be used for inter-library loans and librarians may not upload Springer Nature journal content on a large scale into their, or any other, institutional repository.
These terms of use are reviewed regularly and may be amended at any time. Springer Nature is not obligated to publish any information or content on this website and may remove it or features or functionality at our sole discretion, at any time with or without notice. Springer Nature may revoke this licence to you at any time and remove access to any copies of the Springer Nature journal content which have been saved.
To the fullest extent permitted by law, Springer Nature makes no warranties, representations or guarantees to Users, either express or implied with respect to the Springer nature journal content and all parties disclaim and waive any implied warranties or warranties imposed by law, including merchantability or fitness for any particular purpose.
Please note that these rights do not automatically extend to content, data or other material published by Springer Nature that may be licensed from third parties.
If you would like to use or distribute our Springer Nature journal content to a wider audience or on a regular basis or in any other manner not expressly permitted by these Terms, please contact Springer Nature at

onlineservice@springernature.com