A Multiple Data Collection Tree Protocol for UWSNs

Khaled Day[1], Faiza Al-Salti[2], Nasser Alzeidi[1], and Abderezak Touzene1[1]
[1] Department of Computer Science, Sultan Qaboos University, Muscat 123, Oman
[2] Sultan Qaboos Comprehensive Cancer Care and Research Center, Muscat, Oman
Email: {kday; alzidi; touzene}@squ.edu.om; f.alsalti@cccrc.gov.om

Abstract—We propose a new efficient and robust routing protocol for underwater wireless sensor networks (UWSNs) called the Multiple Data Collection Tree (MDCT) protocol. MDCT proactively constructs and maintains multiple node-disjoint shortest-path routing trees connecting the underwater sensor nodes to onshore sink nodes. These trees provide readily available paths for routing data packets from underwater sensor nodes to surface sink nodes. Using multiple trees improves reliability, reduces congestion (especially at near-root nodes), and shortens routing paths. It also balances energy consumption by distributing the packet-forwarding load over a larger number of nodes. MDCT updates the trees continuously in response to changing underwater conditions such as sensor movements (due to underwater currents) and sensor out-of-power failures. We prove formally the correctness and optimality of the constructed trees. We also show how MDCT outperforms other protocols (namely, VBF, ERGR-EMHC and DCTP) in terms of packet delivery ratio, average end-to-end delay and energy consumption via extensive simulation. For example, compared to VBF, MDCT has increased the delivery ratio by over 75%, has reduced the average end-to-end delay by nearly 60%, and has reduced the energy consumption by 25% in some tested scenarios.

Index Terms—Underwater wireless sensor networks, routing protocols, data collection, multiple disjoint trees, fault-tolerance.

I. Introduction

Underwater Wireless Sensor Networks (UWSNs) enable the exploration of the underwater environment of oceans and seas. There exist different types of architectures for UWSNs but in general, an UWSN consists of a set of sensor nodes deployed under the water surface. These sensors collect different types of data and forward it to onshore sink nodes positioned at the water surface. The collected data may then be transferred to remote data centers or offshore base stations for further processing and analysis [1], [2]. UWSNs are increasingly becoming a potential enabler in multiple application domains including environment monitoring, exploration of natural resources, disaster detection and early warning, and security and military surveillance [3].

Acoustic communication is the only practical communication medium underwater due to short propagation ranges and high absorption of electromagnetic waves. The use of acoustic communication however poses multiple challenges due to high propagation delay, low bandwidth, and high noise and path loss resulting in high error rates [2], [4], [5]. In addition to these challenges related to the communication medium, designs of UWSNs should also take into consideration the harsh underwater environment, the high cost of devices and logistics, the unavailability of accurate positioning techniques and last but not least the stringent energy requirements. These challenges have attracted the attention of researchers to address many related problems including deployment strategies, reliable communication, routing, medium access, localization and energy conservation [1], [6]-[13].

Routing in UWSNs over multiple hops is more efficient than using single long hops [2], [4]. Several routing protocols have been proposed. Vector-based routing protocols [14] construct a virtual pipe between source and destination for packet forwarding. The width of the virtual pipe affects the performance of these protocols. Grid-based routing protocols [10], [15]-[17] is another class of protocols, which divide the area into 2D or 3D grid cells and the routing determines the sequence of cells to traverse. In each cell, a single node acts as a cell-head responsible of forwarding packets across the cell. For example, in the Efficient and Reliable Grid-based Routing by Exploiting Minimum Hop Count (ERGR-EMHC) protocol [17], the election of the cell-heads depends on the residual energy of the nodes and their distances to the center of the cell. The performance of these protocols is highly affected by the node mobility and the density of the network. Other protocols have emerged recently based on building a routing tree for forwarding the collected data from sensor nodes to a sink node [18], [19]. These tree-based protocols are suitable for sparse deployments of sensor nodes. They also tend to consume less energy since only one forwarder relays a packet at each routing step. However, the use of a single tree has the disadvantages of low fault-tolerance and high load on near-root nodes.

We extend this class of tree-based protocols by proposing a multiple data collection tree (MDCT) protocol for routing collected data packets from sensor nodes to a set of onshore sink nodes and then to an offshore base station. MDCT builds and maintains multiple node-disjoint shortest-path trees connecting the sensor nodes to the onshore sink nodes. It routes each data packet to the closest onshore sink node, which then relays it to the offshore base station as illustrated in Fig. 1. Acoustic signals are used for underwater communication while radio communication can be used to relay the collected

Manuscript received June 29, 2021; revised January 14, 2022.
This work was supported by Sultan Qaboos University Grant No. IG/SCI/COMP/19/01.
Corresponding author email: kday@squ.edu.om
doi:10.12720/jcm.17.2.90-98

data from the onshore sink nodes to the offshore base station. MDCT is an improvement of the previously proposed single tree based DCTP protocol [19]. MDCT constructs and uses multiple node-disjoint trees for forwarding packets instead of using a single tree. This improves reliability (fault-tolerance), reduces congestion, and shortens routing paths. It also balances energy consumption by distributing the packet-forwarding load over more near-root nodes. This in turn has substantial positive impact on the delivery ratio, end-to-end delay and energy consumption as confirmed by the obtained simulation results. MDCT updates regularly the trees' links adapting to changing nodes positions (due to underwater currents) and nodes availability (due for example to node failures).

The rest of the paper is organized as follows: Section 2 describes the proposed MDCT protocol. Section 3 presents proofs of correctness and optimality of the constructed trees. Section 4 presents and discusses obtained simulation results and Section 5 concludes the paper.

II. The MDCT Protocol

The objective of the MDCT protocol is to route efficiently the data packets generated at the sensor nodes to the offshore base station via the onshore sink nodes. MDCT aims to route each data packet to the nearest onshore sink node using the least number of routing hops. In order to attain this goal, MDCT constructs and maintains up-to-date multiple node-disjoint shortest-path trees rooted at the onshore sink nodes (see Fig. 1) and uses the trees' links for routing the generated data packets. The MDCT protocol outlined in Fig. 2 carries out the following two functions: (a) the construction and maintenance of the multiple node-disjoint data collection trees; and (b) the forwarding of data packets over these trees.

We describe in the following these two MDCT functions making use of the notations defined in Table I.

TABLE I: NOTATIONS

BS	the offshore base station
k	number of onshore sink nodes
s	an onshore sink node, $s \in \{1, 2, \ldots, k\}$
T_s	data collection tree rooted at s
n	number of underwater sensor nodes
x	underwater sensor node, $x \in \{1, 2, \ldots, n\}$
π_x	parent of sensor node x in its current tree T_s
τ	tree updating period
L_x	level of node x in its current tree (root is at level 0)
Q_x	current sequence number at node x
$BEACON$	a beacon sent by y (y is BS, a sink, or a sensor node)
$<y, Q_y, L_y>$	containing y's sequence number Q_y and its level L_y
$\pi\text{-}set_x$	a set of alternative parent nodes for sensor node x

A. Multiple Trees Construction and Maintenance

MDCT constructs and periodically updates multiple trees denoted T_s's each rooted at an onshore sink node s. Each tree-updating period (marked by a new sequence number) is initiated by the offshore base station BS. At the start of each period (every τ seconds), BS increments its local sequence number Q_{BS} and sends to each of the k sink nodes a beacon packet $BEACON<BS, Q_{BS}, L_{BS}>$. This beacon packet contains the incremented Q_{BS} value as well as the special level value $L_{BS} = -1$ (indicating that BS does not belong to any of the T_s trees). Each sink node s is considered at level 0 (root level) in its own tree T_s. Upon receiving this beacon packet from the base station BS, an onshore sink node s, $s \in \{1, 2, \ldots, k\}$, updates its sequence number (i.e. sets Q_s to Q_{BS}) provided that Q_{BS} is more recent than Q_s (i.e., $Q_{BS} > Q_s$) and sends a beacon packet $BEACON<s, Q_s, L_s>$ to all sensor nodes in its transmission range. If, $Q_{MS} \leq Q_s$, then s discards the beacon packet.

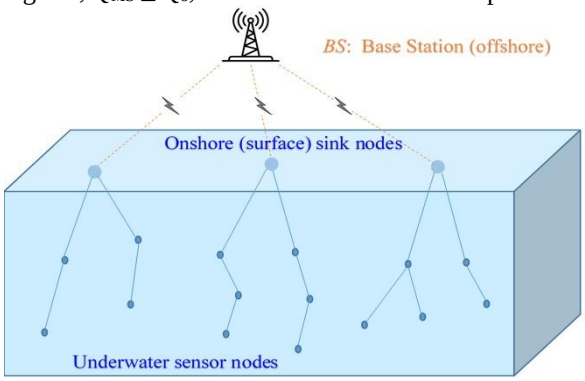

Fig. 1. Multiple (node-disjoint) data collection trees

At any given time, each underwater sensor node x, $x \in \{1, 2, \ldots, n\}$, is attached to at most one T_s tree (Fig. 1). The parent of a sensor node x in its current T_s tree is maintained in a local variable called π_x. For any x, π_x is initially null (initially the node is not attached to any tree). The level of a sensor node x in its current tree T_s (number of hops from x to the root s) is maintained in a local variable called L_x. Each sensor node x also maintains a local sequence number variable called Q_x initialized to zero and updated upon receiving a beacon packet. At any given time, the value of Q_x indicates the largest sequence number that has been seen by node x so far. This corresponds to the sequence number of the latest tree update beacon that has propagated from BS to the node x. Node x also maintains a variable called $\pi\text{-}set_x$ which is a set of sensor nodes. Each node y in $\pi\text{-}set_x$ is such that y is within transmission range of x, and y is one hop closer to a sink node than x (i.e. $L_y = L_x\text{-}1$ for each $y \in \pi\text{-}set_x$). This set represents a set of alternative parent nodes all at the same number of hops from onshore sink nodes (one hop closer to sink nodes than node x). For load balancing, the node x selects its parent π_x randomly from this set $\pi\text{-}set_x$ when a new alternative parent is identified.

When a node x receives a $BEACON<y, Q_y, L_y>$ beacon packet from a node y (y could be either an onshore sink node or another sensor node), it processes it as follows:

- Case 1: If (π_x = null)

This means node x is currently not attached to any tree. In this case, it attaches itself to the tree to which the sender y is attached. This is done by the *New_Parent* (y, Q_y, L_y)

function, which sets π_x to y, Q_x to Q_y and L_x to L_y+1. It also sets the set of alternative parent nodes π-set_x to $\{y\}$.

Offshore Base Station (BS):
- initialize Q_{BS} to 0
- set level L_{BS} to -1
- every period (every τ seconds)
  - $Q_{BS} = Q_{BS} + 1$
  - send BEACON<BS, Q_{BS}, L_{BS}> to sink nodes
- when a data PACKET is received from a sink:
  - process PACKET

Onshore Sink Node s:
- initialize Q_s to 0
- set level L_s to 0
- when BEACON<BS, Q_{BS}, L_{MS}> is received from BS:
  - if($Q_{BS} > Q_s$)
    - $Q_s = Q_{BS}$
    - send BEACON<s, Q_s, L_s> to all sensor nodes in range
- when a data PACKET is received from a sensor:
  - send PACKET to the base station BS

Sensor Node x:
- initializations:
  - $Q_x = 0$
  - π_x = NULL
  - π_set_x = empty
- when x receives a BEACON<y, Q_y, L_y> from y
  - if π_x = NULL //node x has currently no parent
    - call New_Parent (y, Q_y, L_y)
  - else if ($Q_y > Q_x$) //a more recent period
    - call New_Parent (y, Q_y, L_y)
  - else if ($Q_y = Q_x$)
    - if ($L_y < L_x$-1) // y is closer to a sink
      - call New_Parent (y, Q_y, L_y)
    - else if ($L_y = L_x$-1) // alternative parent
      - call Alternative_Parent (y, Q_y, L_y)
  - if no beacon sent in last τ seconds then
    - send BEACON<x,Qx,Lx> to sensors in range
- when x wants to send a data PACKET to a sink
  - if ($\pi_x \neq$ NULL)
    - send PACKET to π_x
  - else discard PACKET // cannot forward packet
- when x receives a data PACKET (for forwarding)
  - if ($\pi_x \neq$ NULL)
    - send PACKET to π_x
  - else discard PACKET // cannot forward packet
- function New_Parent(y, Q_y, L_y) {
  - //set y as the new parent
  - $\pi_x = y$
  - $Q_x = Q_y$
  - $L_x = L_y + 1$
  - $\pi\_set_x = \{y\}$
- }
- function Alternative_Parent(y){
  - //add to alternative parents set
  - $\pi\_set_x = \pi\_set_x \cup \{y\}$
  - //select another parent from parent set for load balancing
  - π_x = Random_Select(π_set_x)
- }

Fig. 2. Overview of the MDCT protocol

- Case 2: If ($\pi_x \neq$ null) and ($Q_y > Q_x$)

This means the beacon is associated with a more recent tree update period than previously seen by the node x. In order to deal with the dynamic nature of the connections between the nodes due to node mobility and node failures, a more recent period always overrides previous periods. Therefore, x performs the same actions as in Case 1.

- Case 3: If ($\pi_x \neq$ null) and ($Q_y = Q_x$) and ($L_y < L_x - 1$)

This means the beacon is associated with the same tree update period as the one most recently seen by the local node, but the level of the sender y is smaller than the level of the current parent of the local node (which is $L_x - 1$). Hence, the sender is closer to an onshore sink node than the current parent of x is. In this case, node x performs the same actions as in cases 1 and 2.

- Case 4: If ($\pi_x \neq$ null) and ($Q_y = Q_x$) and ($L_y = L_x - 1$)

This means the beacon is associated with the same tree update period as the one last seen by the local node. It also means that the level of the sender y is equal to the level of the current parent of x (which is equal to $L_x - 1$). In this case the sender y is considered as an alternative parent and is therefore added to the set π-set_x of alternative parents of node x. The parent π_x of x is then updated by selecting randomly one of the nodes in π-set_x for load balancing. These actions are performed by invoking the function *Alternative_Parent* (y, Q_y, L_y).

- Case 5: If ($\pi_x \neq$ null) and ($Q_y < Q_x$)

In this case, x discards the received beacon since it is associated with an older update period than last seen by x.

After processing a received beacon, node x forwards a beacon packet *BEACON*<x, Q_x, L_x> to all sensor nodes in its transmission range if it has not done so in the last τ seconds, otherwise it refrains from forwarding this beacon. This ensures the regular updating of the trees aiming at providing the shortest possible routing paths, without sending too many beacons unnecessarily.

B. Forwarding of Data Packets in MDCT

The routing of data packets from sensor nodes to onshore sink nodes uses the constructed data collection trees. When a sensor node generates a data packet, it sends it to its current parent node π_x if it is not null. If, however, this latter is null, then it discards the data packet. Likewise, when the node receives a data packet for forwarding. It forwards it to its parent, which will forward it to its parent and so on until it reaches an onshore sink node, which forwards it to the offshore base station *BS*.

III. Trees' Correctness and Optimality

In this section, we show that the node-parent relation established by the MDCT protocol outlined in Fig. 2 defines a set of node-disjoint directed trees each having an onshore sink node as a sink (a vertex without outgoing edges). This implies that MDCT routes data packets correctly from sensor nodes to onshore sink nodes over these trees without looping. We also show that each sensor node is attached to the tree whose sink is the nearest onshore sink node. We derive from this property that a data packet is routed along a shortest path from the source sensor node to the nearest onshore sink node.

Definition 1: Let s be a sink node $s \in \{1, 2, \ldots, k\}$. We define $T_s = (V_s, E_s)$ as the directed graph given by:

$V_s = \{s\} \cup \{\text{sensor } x \mid \pi_x = s\} \cup$
$\{\text{sensor } x \mid \exists \text{ sensors } y_1, y_2, \ldots, y_h \text{ such that } \pi_x = y_1, \pi_{y_i} = y_{i+1}, 1 \le i < h \text{ and } \pi_{y_h} = s\}$

$E_s = \{(x, y) \text{ such that } x \in V_s, y \in V_s \text{ and } \pi_x = y\}$

Notice that the set of vertices V_s of T_s consists of the onshore sink node s and any sensor node x for which there is a node-to-parent sequence (path) leading from x to s. The edges of T_s represent node-to-parent links. We shall prove that at any time, T_s is a directed tree with sink s. We first establish the following two lemmas.

Lemma 1: For any node x at any time, if $y \in \pi\text{-set}_x$ then $(Q_y > Q_x)$ or $(Q_y = Q_x$ and $L_y < L_x)$.

Proof: We prove that for any node x, the property is initially true and that it remains true after any modification that affects the variables in this property assuming that the property was true before the modification. The only modifications that can affect the property are those that modify: (a) $\pi\text{-set}_x$, (b) Q_y, for any $y \in \pi\text{-set}_x$, (c) Q_x, (d) L_y, for any $y \in \pi\text{-set}_x$, or (e) L_x. Since $\pi\text{-set}_x$ is initially set to empty, the claimed property is therefore initially vacuously true. Now we show that the property remains true after any of the modifications (a) to (e) assuming it was true before the modification.

(a) $\pi\text{-set}_x$ is modified by the MDCT protocol only in the following four situations:
  (i) $\pi\text{-set}_x$ is modified when x receives a beacon from a node y at a time when π_x = null. In this case, node x calls New_Parent(y, Q_y, L_y) which sets π_x to y, Q_x to Q_y, L_x to L_y+1 and $\pi\text{-set}_x$ to $\{y\}$. After these settings the property is true since $Q_y = Q_x$ and $L_y < L_x$.
  (ii) $\pi\text{-set}_x$ is also modified when x receives a beacon from a node y with $Q_y > Q_x$. As in the previous case (a)-(i), in this case node x calls the function New_Parent(y, Q_y, L_y) which sets π_x to y, Q_x to Q_y, L_x to L_y+1 and $\pi\text{-set}_x$ to $\{y\}$. After these settings the claimed property is true since $Q_y = Q_x$ and $L_y < L_x$.
  (iii) $\pi\text{-set}_x$ is also modified when x receives a beacon from a node y at a time when $Q_x = Q_y$ and $L_y < L_x$-1. As in case (a)(i), in this case x calls the function New_Parent(y, Q_y, L_y) which sets π_x to y, Q_x to Q_y, L_x to L_y+1 and $\pi\text{-set}_x$ to $\{y\}$. After these settings the claimed property is true since $Q_y = Q_x$ and $L_y < L_x$.
  (iv) Finally, $\pi\text{-set}_x$ is modified when x receives a beacon from a node y at a time when $Q_x = Q_y$ and $L_y = L_x$-1. In this case, node x calls the function Alternative_Parent(y, Q_y, L_y) which adds y to $\pi\text{-set}_x$,. Q_x remains equal to Q_y, and L_x remains equal to L_y+1. After these settings, the property remains true since for the only added node y to $\pi\text{-set}_x$, we have $Q_y = Q_x$ and $L_x = L_y$+1 (hence $L_y < L_x$). No other node z in $\pi\text{-set}_x$ is affected and hence the claimed property $(Q_z > Q_x)$ or $(Q_z = Q_x$ and $L_z < L_x)$ remains true since it was true before the modification and neither Q_x nor L_x have been modified.

(b) For any $y \in \pi\text{-set}_x$, Q_y is only modified when node y receives a beacon packet from a node z with $Q_z > Q_y$. In this case, Q_y is set to a higher value (namely Q_z). Since the property was true before the modification, then we must have had either $(Q_y > Q_x)$ or $(Q_y = Q_x$ and $L_y < L_x)$ before the modification. In both cases, we will have $Q_y > Q_x$ after the modification since Q_y is set to a higher value Q_z. Hence, the property remains true.

(c) Q_x is only modified when node x receives a beacon packet from a node y with $Q_y > Q_x$. In this case, Q_x is set to Q_y and L_x is set to L_y+1. Therefore, after the modification we have $Q_y = Q_x$ and $L_y < L_x$.

(d) For any $y \in \pi\text{-set}_x$, L_y is modified only in the following two situations:
  (i) L_y is modified when node y receives a beacon packet from a node z with $Q_z > Q_y$. In this case, Q_y is set to a higher value (namely Q_z). Therefore, we will have $Q_y > Q_x$ after these modifications for the same reasons given in case (b). Hence, the property remains true.
  (ii) L_y is also modified when y receives a beacon from a node z with $Q_z = Q_y$ and $L_z < L_y$-1. In this case, Q_y is not modified and L_y is set to a smaller value L_z+1 (since $L_z < L_y$-1). Since Q_y is not modified and L_y is decreased, $(Q_y > Q_x)$ or $(Q_y = Q_x$ and $L_y < L_x)$ remains true, if it was true before the modification.

(e) L_x is modified only in the following two situations:
  (i) L_x is modified when node x receives a beacon packet from a node y with $Q_y > Q_x$. As in case (a)-(i), node x calls the function New_Parent(y, Q_y, L_y) which sets π_x to y, Q_x to Q_y, L_x to L_y+1 and $\pi\text{-set}_x$ to $\{y\}$. After these settings the claimed property is true since $Q_y = Q_x$ and $L_y < L_x$.
  (ii) L_x is also modified when x receives a beacon from a node y with $Q_y = Q_x$ and $L_y < L_x$-1. As in case (a)-(i), node x calls the function New_Parent(y, Q_y, L_y) which sets π_x to y, Q_x to Q_y, L_x to L_y+1 and $\pi\text{-set}_x$ to $\{y\}$. After these settings the claimed property is true since $Q_y = Q_x$ and $L_y < L_x$.

Hence, the claimed property remains true after any of the modifications (a) to (e). QED

Lemma 2: For any sensor x at any time, if $\pi_x = y$, then $(Q_y > Q_x)$ or $(Q_y = Q_x$ and $L_y < L_x)$.

Proof: For any sensor node x, π_x is initially null and hence the claimed property is initially vacuously true. Subsequently, π_x is only modified when node x invokes New_Parent( ) or Alternative_Parent( ). From Fig. 2, one can easily verify that after executing any of these two functions, we have $\pi_x \in \pi\text{-set}_x$. Therefore, based on Lemma 1, the claimed property is satisfied. QED

Proposition 1: For any sink node $s \in \{1, 2, \ldots, k\}$, T_s is a directed tree with sink s (hence for any node x in T_s, there exists a directed path from x to s).

Proof: For any sink node $s \in \{1, 2, \ldots, k\}$, let R_s be the binary relation on the set V_s defined as follows: $x\ R_s\ y$ if, and only if, $(Q_y > Q_x)$ or $(Q_y = Q_x$ and $L_y < L_x)$. We prove that R_s is a strict partial order on the set V_s and then derive from this property that T_s is acyclic. To prove that R_s is a strict partial order, we have to show that (a) R_s is irreflexive, (b) R_s is transitive, and (c) R_s is asymmetric.

(a) R_s *is irreflexive:* Assume $x\ R_s\ y$ for some nodes x and y in V_s. Then ($Q_y > Q_x$) or ($Q_y = Q_x$ and $L_y < L_x$). Therefore, $x \neq y$. Hence, R_s is irreflexive.

(b) R_s *is transitive:* Assume $x\ R_s\ y$ and $y\ R_s\ z$ for some nodes x, y and z in V_s. Since $x\ R_s\ y$, then ($Q_y > Q_x$) or ($Q_y = Q_x$ and $L_y < L_x$). Since also $y\ R_s\ z$, then ($Q_z > Q_y$) or ($Q_z = Q_y$ and $L_z < L_y$). There are four cases:
   (i) If ($Q_y > Q_x$) and ($Q_z > Q_y$), then $Q_z > Q_x$ and hence $x\ R_s\ z$.
   (ii) If ($Q_y > Q_x$) and ($Q_z = Q_y$ and $L_z < L_y$), then $Q_z > Q_x$ and hence $x\ R_s\ z$.
   (iii) If ($Q_y = Q_x$ and $L_y < L_x$) and ($Q_z > Q_y$), then $Q_z > Q_x$ and hence $x\ R_s\ z$.
   (iv) If ($Q_y=Q_x$ and $L_y<L_x$) and ($Q_z=Q_y$ and $L_z<L_y$), then ($Q_z=Q_x$ and $L_z<L_x$) hence $x\ R_s\ z$.
   Therefore, $x\ R_s\ z$ in all cases. Hence, R_s is transitive.

(c) R_s *is asymmetric:* Assume $x\ R_s\ y$ for x and y in V_s. Then ($Q_y > Q_x$) or ($Q_y = Q_x$ and $L_y < L_x$). If $Q_y > Q_x$ then neither $Q_x > Q_y$ is true nor $Q_x = Q_y$ is true. Therefore, $y\ R_s\ x$ is not true. If ($Q_y = Q_x$ and $L_y < L_x$) then neither $Q_x > Q_y$ is true nor $L_x < L_y$ is true and therefore, $y\ R_s\ x$ is not true in this case too. Hence, R_s is asymmetric.

Therefore, R_s is a strict partial order on V_s. Furthermore, by Definition 1, for any edge (x, y) in E_s, we have $\pi_x = y$ and hence by Lemma 2, $x\ R_s\ y$. Therefore, T_s cannot possibly contain any cycles. Therefore, T_s is a directed acyclic graph (DAG). In addition, by Definition 1, for any node x in V_s, either x is s or there exists a directed path in T_s from x to s. Therefore, T_s is connected. Since T_s is connected and is acyclic, then T_s is a tree. Since in addition s has no outgoing edge in T_s, we conclude that T_s is a directed tree with sink s. QED

Proposition 2: For any two distinct sink nodes s_1 and s_2, T_{s_1} and T_{s_2} are node-disjoint.

Proof: Assume there exists a common node x to T_{s_1} and T_{s_2}. Since s_1 does not belong to T_{s_2} and s_2 does not belong to T_{s_1}, we must have $x \neq s_1$ and $x \neq s_2$. So, x can only be a common sensor node in T_{s_1} and T_{s_2}. Since x is in T_{s_1}, there must exist a directed path ($x \rightarrow y_1 \rightarrow y_2 \rightarrow \ldots \rightarrow y_h \rightarrow s_1$) from x to s_1 in T_{s_1}. Similarly, since x is in T_{s_2}, there must exist a directed path ($x \rightarrow z_1 \rightarrow z_2 \rightarrow \ldots \rightarrow z_l \rightarrow s_2$) from x to s_2 in T_{s_2}. Let m be the smallest integer such that $y_m \neq z_m$ (m must exist since $s_1 \neq s_2$). If $m = 1$, then $y_1 \neq z_1$ and $\pi_x = y_1$ and $\pi_x = z_1$ which is not possible since x has a unique parent π_x at any given time. If, $m > 1$, then we have $y_{m-1} = z_{m-1}$ and $y_m \neq z_m$. Therefore, we must have $\pi_{y_{m-1}} = y_m$ and $\pi_{y_{m-1}} = z_m$ which is also not possible since y_{m-1} has a unique parent $\pi_{y_{m-1}}$ at any given time. We conclude by contradiction that T_{s_1} and T_{s_2} have no common nodes. Hence, they are node-disjoint. QED

Now we show that after *BS* issues an update beacon, the updated trees will converge to shortest-path trees if any connected nodes (within transmission range of each other) remain connected during the time needed for the update to propagate to all reachable nodes.

Definition 2: At any given time t, any sink node s, and any sensor node x, let $\delta_x^s(t)$ denote the minimum number of hops from x to s.

We assume in the following proposition, that the time needed to process a beacon by any node and the time needed to transmit a beacon from a node to another node in its transmission range are both negligible compared to the length τ of the updating period. We assume τ is large enough for this assumption to be acceptable.

Proposition 3: Assume a sink node s has become the closest sink node to a sensor node x at time t. Therefore, x will be attached to the tree T_s by time $t + \delta_x^s(t)\tau$, provided that any connected nodes (within transmission range of each other) remain so during the period $[t, t + \delta_x^s(t)\tau]$.

Proof: Let x be any sensor node. Assume that at some time t, a sink node s has become the sink node that requires the least number of hops h to reach from x. Let $x \rightarrow y_1 \rightarrow y_2 \rightarrow \ldots \rightarrow y_{h-1} \rightarrow s$ be these h hops starting at node x going through sensor nodes $y_1, y_2, \ldots, y_{h-1}$ in the first h-1 hops and ending at the sink node s in the last hop. We therefore have, $\delta_x^s(t) = h$. By the optimal sub-structure property of the shortest path, sink node s must also be the closest sink node to each y_i, $1 \leq i \leq h$-1. The offshore base station *BS* issues a beacon packet every τ seconds. Hence, BS issues at least one beacon packet during the time interval $[t, t + \tau]$. Furthermore, every onshore sink node (including s) forwards immediately the beacon packet received from *BS* to all sensor nodes in its transmission range. Therefore, the node y_{h-1} receives a beacon from s and attaches itself to T_s (sets s as parent) by time $t+\tau$ (since by the optimal sub-structure property, s is also the closest sink to y_{h-1}). The sensor node y_{h-2} receives a beacon from y_{h-1} and attaches itself to T_s (sets y_{h-1} as its parent) by time $t+2\tau$. This continues until node y_1 receives a beacon from y_2 and attaches itself to T_s (sets y_2 as its parent) by time $t+(h-1)\tau$ and then finally sensor node x receives a beacon from y_1 and attaches itself to T_s (sets y_1 as its parent) by time $t + h\tau$ which is equal to $t + \delta_x^s(t)\tau$. QED

IV. Simulation-Based Performance Evaluation

We have simulated the MDCT protocol using the Aqua-Sim simulator [21] which is an NS2 [22] based simulator. We have also implemented three other protocols, namely VBF [14], ERGR-EMHC [17] and DCTP [19], in Aqua-Sim in order to compare their performance with MDCT's performance. The VBF protocol is selected because it is one of the most widely cited and used in comparison with other routing protocols for UWSNs. The ERGR-EMHC is selected because it is one of the recent routing protocols and DCTP is selected as it is the predecessor of the MDCT protocol.

The size of the simulated underwater area is set to (3×3×3) km^3. The transmission range of the nodes is set to 0.8 km. The transmission, reception and idle powers are set to 8.0 W, 0.80 W and 0.008 W, respectively. The bit error rate is set to 10e-9. The sensor nodes are initially deployed randomly in the 3D simulation area with possible movement with water currents. Sink nodes are deployed at the surface using the Multiple Sinks Placement (MSP) scheme proposed in [20] to minimize the number of hops between each sensor node and its nearest sink node.

The CSMA-based MAC protocol [14], [17], [23], [24] is used. Each simulation experiment runs for 2000 seconds. The selected source nodes inject data packets according to a random exponential distribution. We obtain the average of 25 batch runs and error bars (with 95% confidence). Table II lists the remaining simulation settings.

TABLE II: SIMULATION SETTINGS

Parameter	Value
number of sink nodes	1 or 3
number of nodes	54, 162, 270
initial energy	300 J
data packet size	150 Bytes
traffic injection rate	0.08 packets/sec
sink beacon period (DCTP, MDCT)	R/max speed
sensor beacon period (DCTP)	2 * sink beacon period
maximum speed	(0.1, 0.5, 1.0, 1.5) m/sec
sending Probability	0.3, 0.5, 0.7, 0.9
pipe width (VBF)	400 m
energy threshold (ERGR-EMHC)	10 J
β (ERGR-EMHC)	0.9
iPeriod (ERGR-EMHC)	50 sec
update period (ERGR-EMHC)	500 sec
new election period (ERGR-EMHC)	400 sec

We have used the following performance measures:

- *Packet Delivery Ratio (PDR):* the number of successfully delivered data packets divided by the total number of generated data packets.
- *Average End-to-End Delay:* the average time that takes a successfully delivered data packet to propagate from the source sensor node to a sink node.
- *Energy Consumption:* total energy consumed by all sensor nodes in transmission, reception and idle modes.

We have conducted three sets of experiments to measure the effect on the above three metrics of the number of sensor nodes, the traffic load (packet generation probability) and the maximum node mobility speed.

A. Effect of the Number of Sensor Nodes

In this set of experiments, we have fixed the sending probability to 0.3 (i.e., 30% of the nodes generate traffic) and the maximum node mobility speed to 0.1 m/s.

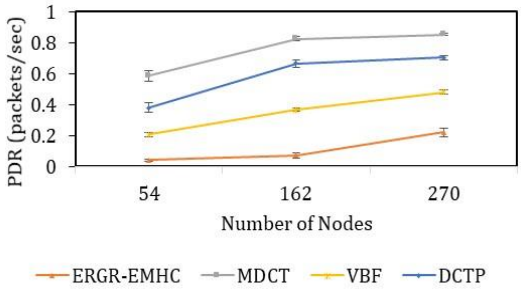

Fig. 3. PDR vs. the number of nodes

Fig. 3 shows the effect of the number of nodes on the packet delivery ratio (PDR). MDCT has achieved the highest PDR in all tested scenarios. For example with 162 sensor nodes, the four protocols MDCT, DCTP, VBF and ERGR-EMHC have delivered successfully 82%, 66%, 37% and 7% of the generated data packets, respectively.

Fig. 4 shows the effect of varying the number of nodes on the end-to-end delay. The average end-to-end delay of both VBF and ERGR-EMHC increases with the increase in the number of sensor nodes while it decreases for DCTP and MDCT. This can be justified by the use of a single forwarding node at each hop in both DCTP and MDCT, which is not the case in ERGR-EMHC and VBF.

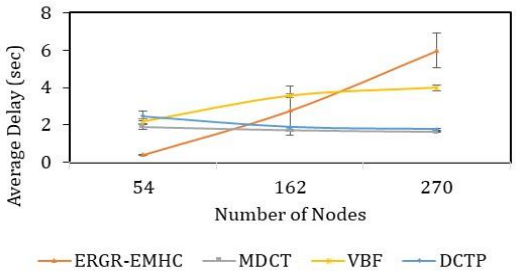

Fig. 4. Average end-to-end delay vs. the number of nodes

With 162 sensor nodes, MDCT has delivered packets faster by 10%, 38%, and 52% than DCTP, ERGR-EMHC and VBF, respectively.

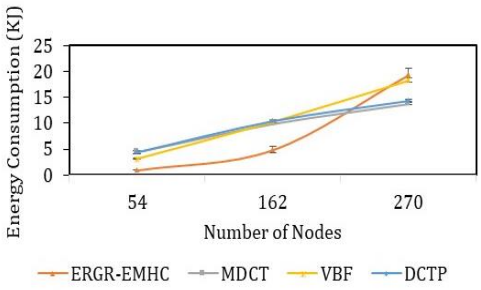

Fig. 5. Energy consumption vs. the number of nodes

Fig. 5 shows that ERGR-EMHC consumes less energy with less nodes compared to the other protocols. However, as the number of nodes increases, DCTP and MDCT outperform both VBF and ERGR-EMHC. This is due to the smaller number of routing hops and the use of a single forwarder in DCTP and in MDCT. Increasing the number of nodes from 54 to 270, has caused an increase in energy consumption by 68%, 69%, 95% and 83%, in MDCT, DCTP, ERGR-EMHC and VBP respectively.

B. Effect of the Traffic Load

In this set of experiments, we have used 162 randomly deployed sensor nodes with a maximum mobility speed of 0.1 m/s. We Vary the traffic load by varying the sending probability (the probability of sending a generated data packet).

Fig. 6 shows that overall, PDR decreases when increasing the sending probability due to the increase in the number of generated data packets in the network. However, MDCT is superior in delivering data packets compared to DCTP, VBF and ERGR-EMHC. For example, when the sending probability is set to 0.9, MDCT outperforms DCTP, VBF and ERGR-EMHC in delivering data packets by nearly 41%, 60% and 88%, respectively.

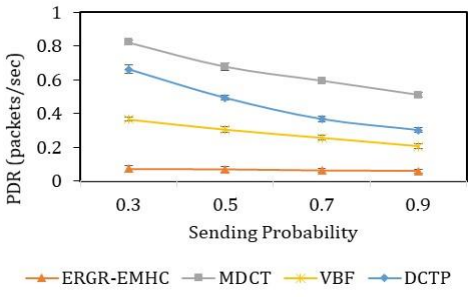

Fig. 6. PDR vs. the traffic load

Fig. 6 also shows how the use of multiple trees in MDCT has reduced congestion and hence increased PDR compared to using a single tree in DCTP.

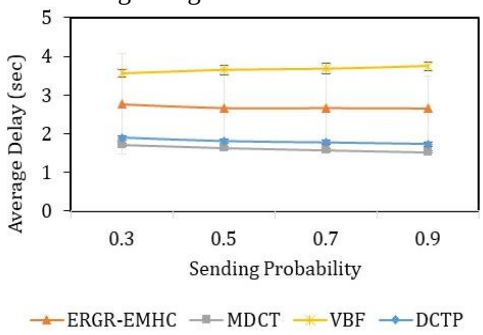

Fig. 7. Average end-to-end delay vs. the traffic load

Fig. 7 shows that for all protocols, varying the sending probability in the range from 0.3 to 0.9 has little effect on the average end-to-end delay. However, MDCT is the fastest in delivering data packets compared to the other protocols. When the sending probability is set to 0.9, MDCT has delivered packets 12%, 52% and 42% faster than DCTP, VBF and ERGR-EMHC, respectively.

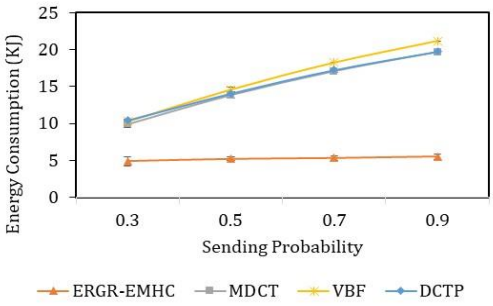

Fig. 8. Energy consumption vs. the traffic load

Fig. 8 shows that increasing the sending probability in the range 0.3 to 0.9 has little effect on the energy consumption of ERGR-EMHC outperforming the other protocols, which consume more energy as the sending probability increases. MDCT consumes less energy than VBF (around 7% saving for a sending probability of 0.9).

C. *Effect of the Nodes' Mobility*

In this set of experiments, the number of sensor nodes is set to 162 with sending probability of 0.3.

Fig. 9 shows that MDCT outperforms the other protocols in terms of PDR for the different node speeds. For instance, when the maximum speed is set to 1m/s, MDCT delivers more data packets than DCTP, VBF and ERGR-EMHC by 15%, 25% and 197%, respectively.

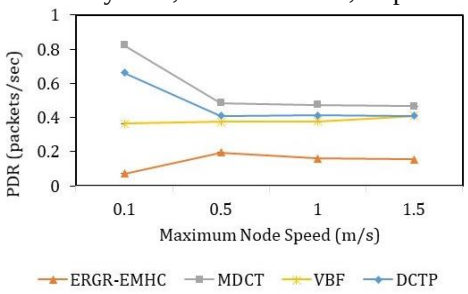

Fig. 9. PDR vs. maximum node speed

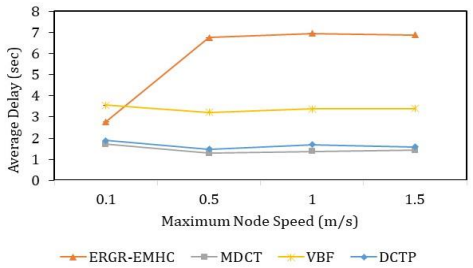

Fig. 10. Average end-to-end delay vs. maximum node speed

Fig. 10 shows the effect of mobility on the delay. Here also MDCT yields the best results compared to the other protocols. For example, with a maximum speed of 1m/s, MDCT delivers data packets faster that DCTP, VBF and ERGR-EMHC by 19%, 60% and 80%, respectively.

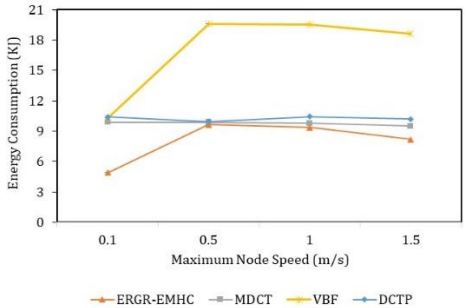

Fig. 11. Energy consumption vs. maximum node speed

Fig. 11 shows the effect of mobility on energy consumption. VBF is the worst in energy consumption while the other three protocols are comparable with a small advantage for ERGR-EMHC over DCTP and MDCT.

Notice from Fig. 9, Fig. 10 and Fig. 11 that overall, increasing the nodes' mobility speed between 0.5m/s and 1.5m/s has a little effect on the performance of MDCT. This is because MDCT updates the node-to-parent links in the trees when nodes move around selecting each time the best (nearest to a sink) node as a parent for each node.

V. Conclusion

The proposed MDCT protocol builds and uses multiple disjoint trees for routing collected data packets from underwater sensor nodes to surface sink nodes. Using multiple trees improves reliability, reduces congestion, and shortens routing paths as compared to using a single

tree. It also balances energy consumption by distributing the traffic load over a larger number of sensor nodes. We presented formal proofs of correctness and optimality of the constructed disjoint trees. Simulation results have shown that MDCT outperforms substantially other protocols with respect to delivery ratio, end-to-end delay and energy consumption. For example, compared to VBF, MDCT has increased the delivery ratio by 77%, has reduced the average end-to-end delay by 59%, and has reduced the energy consumption by 25% in some tested scenarios. The three features of MDCT: (a) using multiple trees instead of a single tree, (b) involving only one forwarding node at each routing step, and (c) updating regularly the trees in response to underwater changing conditions (such as sensor movements with water currents and sensor power failures) with low tree updating cost, have contributed to MDCT's good performance.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

All authors contributed to the discussion of the design of the proposed protocol, reviewing and analyzing the simulation results and writing different sections of the paper; Dr. Faiza Al-Salti conducted the simulation work; all authors reviewed and approved the final version.

REFERENCES

[1] G. Han, C. Zhang, L. Shu, N. Sun, and Q. Li, "A survey on deployment algorithms in underwater acoustic sensor networks," *Int. J. of Dist. Sensor Networks*, pp. 1–11, 2013.

[2] I. Akyildiz, D. Pompili, and T. Melodia, "Underwater acoustic sensor networks: Research challenges," *Ad Hoc Networks*, vol. 3, no. 3, pp. 257–279, 2005.

[3] S. Q. M. Murad, A. A. Sheikh, M. A. Manzoor, and E. Felemban, "A survey on current underwater acoustic sensor network applications," *Int. J. Computer Theory and Engineering*, vol. 7, pp. 51–56, 2015.

[4] Z. Jiang, "Underwater acoustic networks – issues and solutions," *International Journal of Intelligent Control and Systems*, vol. 13, no. 3, pp. 152–161, 2008.

[5] L. Lanbo, Z. Shengli, and C. Jun-Hong, "Prospects and problems of wireless communication for underwater sensor networks," *Wireless Comm. & Mobile Comp.*, vol. 8, no. 8, pp. 977–994, 2008.

[6] B. S. Halakarnimath and A. V. Sutagundar, "Reinforcement learning-based routing in underwater acoustic sensor networks," *Wireless Personal Communications*, vol. 120, pp. 419–446, 2021.

[7] R. W. L. Coutinho, A. Boukerche, L. F. M. Vieira, A. A. F. Loureiro, "A survey of routing protocols for underwater wireless sensor networks," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 137–160, 2021.

[8] H. Khan, S. A. Hassan, and H. Jung, "On underwater wireless sensor networks routing protocols: A review," *IEEE Sensors J.*, vol. 20, no. 18, pp. 10371-10386, 2020.

[9] M. Li, X. Du, X. Liu, and C. Li, "Shortest path routing protocol based on the vertical angle for underwater acoustic networks," *J. of Sensors*, vol. 2019, no. 4, 2019.

[10] F. A. Salti, N. Alzeidi, and B. Arafeh, "EMGGR: An energy-efficient multipath grid-based geographic routing protocol for underwater wireless sensor networks," *Wireless Networks*, vol. 23, no. 4, pp. 1301-1314, 2017.

[11] K. Chen, M. Ma, E. Cheng, F. Yuan, and W. Su, "A survey on MAC protocols for UWSNs," *IEEE Comm. Surveys & Tut.*, vol. 16, no. 3, pp. 1433-1447, 2014.

[12] G. Han, C. Zhang, L. Shu, and J. Rodrigues, "Impacts of deployment strategies on localization performance in underwater acoustic sensor networks," *IEEE Trans. on Industrial Electronics*, vol. 62, no. 3, pp. 1725–1733, 2015.

[13] M. Boulaiche and L. B. Medjkoune, "EGGR: Energy-aware and delivery guarantee geographic routing protocol," *Wireless Net.*, vol. 21, no. 6, pp. 1765–1774, 2015.

[14] P. Xie, J. H. Cui, and L. Lao, "VBF: Vector-based forwarding protocol for underwater sensor networks," in *Proc. of IFIP Net.'06*, vol. 3976, 2006, pp. 1216–1221.

[15] F. Al-Salti, N. Alzeidi, K. Day, B. Arafeh, and A. Touzene, "Grid based priority routing protocol for UWSNs," *Int. J. Comput. Networks Commun.*, vol. 9, no. 6, pp. 1–20, 2017.

[16] K. Wang, H. Gao, X. Xu, J. Jiang, and D. Yue, "An energy-efficient reliable data transmission scheme for complex environmental monitoring in underwater acoustic sensor net.," *IEEE Sens. J.*, vol. 16, no. 11, pp. 4051–4062, 2016.

[17] F. Al-Salti, N. Alzeidi, K. Day, and A. Touzene, "An efficient and reliable grid-based routing protocol for UWSNs by exploiting minimum hop count," *Computer Networks*, vol. 162, 2019.

[18] K. Day, H. Al-Moqbali, N. Alzeidi, and A. Touzene, "TBR: Tree-based routing over a 3D grid for underwater wireless sensor networks," *Jocm*, vol. 12, no. 10, pp. 579–584, 2017.

[19] K. Day, F. Al-Salti, A. Touzene, and N. Alzeidi, "An efficient data collection protocol for UWSNs," *Int. J. of Comp. Networks and Comm.*, vol. 12, no. 5, pp. 1-15, 2020.

[20] F. Al-Salti, K. Day, N. Alzeidi, and A. Touzene, "Multiple sink placement strategy for underwater wireless sensor networks," in *Proc. Int'l Symp. on Networks, Computers and Comm., IEEE ISNCC*, Rome, 2018, pp. 1-6.

[21] P. Xie, *et al.*, "Aqua-Sim: An NS-2 based simulator for underwater sensor networks," in *Proc. MTS/IEEE Biloxi - Marine Technology for Our Future: Global and Local Challenges (OCEANS 2009)*, 2009, pp. 1–7.

[22] T. Issariyakul, E. Hossain, T. Issariyakul, and E. Hossain, "Introduction to network simulator 2 (NS2)," in *Intro. to Network Simulator NS2*, Springer US, 2012, pp. 21–40.

[23] N. Nicolaou, A. See, and P. Xie, "Improving the robustness of location-based routing for underwater sensor networks," *Proceedings of the OCEANS 2007 - Europe*, IEEE, UK, June 2007, pp. 1–6.

[24] P. Xie, Z. Zhou, Z. Peng, J.-H. Cui, and Z. Shi, "Void avoidance in three-dimensional mobile underwater sensor networks," in *Wireless Algorithms, Systems, and Applications*, Springer, 2009, pp. 305–314.

Khaled Day completed his undergraduate studies in Tunisia in 1986. He received the MSc and PhD degrees from the University of Minnesota (USA) in 1989 and 1992, respectively. He is currently Professor at the Department of Computer Science at Sultan Qaboos University. His research interests are in the areas of parallel and distributed computing and networks. He is a senior member of IEEE.

Faiza Al-Salti received the BSc, MSc and PhD degrees in computer science from Sultan Qaboos University in 2012, 2015 and 2019, respectively. She worked as a Lecturer at the University of Technology and Applied Sciences, Oman, and then joined in 2021 the Sultan Qaboos Comprehensive Cancer Care and Research Center. Her research interests include communication protocols and terrestrial and underwater wireless sensor networks.

Nasser Alzeidi received his PhD degree in computer science from the University of Glasgow (UK) in 2007. He is currently an Associate Professor of computer science and the Director of the Center for Information Systems at Sultan Qaboos University. His research interests include performance evaluation of communication systems, wireless networks, interconnection networks, System on Chip architectures and parallel and distributed computing. He is a member of the IEEE.

Abderezak Touzene received his BS from the University of Algiers in 1987, the M.Sc. degree from Orsay Paris-Sud University in 1988 and the PhD degree from the Institute Polytechnique de Grenoble (France) in 1992. He is Professor at the Department of Computer Science at Sultan Qaboos University. His research interests include Cloud Computing, Parallel and Distributed Computing, Wireless and Mobile Networks, Network on Ship (NoC), Cryptography and Network Security, Interconnection Networks, Performance Evaluation, Numerical Methods. He is a member of the IEEE.