SHARK-IA: An Interference Alignment Algorithm for Multi-hop Underwater Acoustic Networks with Large Propagation Delays

Huacheng Zeng[†] Y. Thomas Hou[†] Yi Shi[†] Wenjing Lou[†] Sastry Kompella[‡] Scott F. Midkiff[†]

† Virginia Polytechnic Institute and State University, USA
‡ U.S. Naval Research Laboratory, Washington, DC, USA

ABSTRACT

A fundamental issue of underwater acoustic (UWA) communications is large propagation delays due to water medium, which has posed a big challenge to improving the performance of UWA networks. A new direction to address this issue is to take advantage of large propagation delays rather than enduring them as a disadvantage. Recent advances in time-based interference alignment (IA), or propagation delay (PD)-based IA, promise a great potential to turn the adverse effect of large propagation delays into something that is beneficial to throughput improvement. The goal of this paper is to investigate PD-IA in a *multi-hop* UWA network. We develop an analytical PD-IA model with a set of constraints that guarantee PD-IA feasibility at the physical layer. Based on this model, we develop a distributed PD-IA scheduling algorithm, called SHARK-IA, to maximally overlap interference in a multi-hop UWA network. Simulation results show that SHARK-IA algorithm can improve throughput performance when compared to an idealized benchmark algorithm with perfect scheduling and zero propagation delays. Further, the throughput gain increases with the volume of interference in the network.

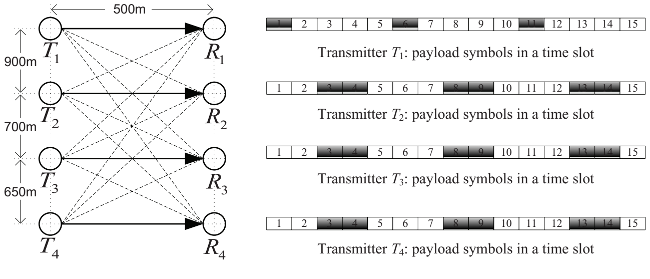
1. INTRODUCTION

The growing need for oceanographic data collection, remote sensing, and tactical communications has led to a surge of research efforts in the area of underwater acoustic (UWA) networks [1]. A fundamental issue in UWA networks is the large propagation delays associated with slow speed of sound when traveling in water (1500 m/s) [18]. For instance, it takes about 2/3 seconds for sound to travel 1 km in water. Such large propagation delay poses a major barrier on the performance of UWA networks.

In the wireless networking community, there have been active research efforts to design efficient algorithms and protocols that took into account large propagation delays in UWA networks [7, 8, 10, 14, 15, 16, 17]. Most of these efforts considered the large propagation delay issue and attempted to develop protocols and algorithms that work around it. In this paper, instead of considering large propagation delays as an adversary, we exploit propagation delays as an advantage for throughput improvement in UWA networks. This is accomplished by time-based interference alignment (IA) technique, which we call propagation delay based IA (PD-IA). PD-IA exploits large propagation delays so that interference from different transmitters can overlap in the same time intervals, allowing more time intervals for data transmission.

1.1 PD-IA: A Motivating Example

To see how PD-IA works in UWA networks with large propagation delays, consider a 4-link network shown in Fig. 1(a), where the solid lines represent intended links and dashed lines represent interference. The propagation delays between any transmit and receive nodes can be computed based on the distances between the nodes and the speed of sound in water (1500 m/s). Suppose that the data transmission is done on time slots, with each time slot carrying 15 OFDM symbols and each OFDM symbol having 85.5 ms duration [12]. Then the normalized propagation delays (with respect to the OFDM symbol duration time) between the transmitters and receivers can be computed, as shown in Table 1. Suppose that we schedule OFDM symbol payload at each transmitter as in Fig. 1(b), where the shadowed intervals represent payload while empty intervals represent idle time. Then at each receiver, it receives its desired symbol stream, plus three interfering symbol streams. Based on their respective propagation delays in Table 1, the received symbol streams at each receiver are shown in Fig. 2. We can see that the desired symbol stream at each receiver is completely separated (no overlap) from the other three interfering streams, thanks to the payload scheduling in Fig. 1(b) in conjunction with the underlying propagation delays between the nodes. For example, in Fig. 2(a), the desired symbols at receiver R_1 (from T_1) are completely free of interference from the other three interfering streams. On the other hand, there is overlap among the symbols from the three interfering streams (T_2, T_3, and T_4). Likewise, similar separation and alignment of desired and interfering symbols occur at receivers R_2, R_3, and R_4, as shown

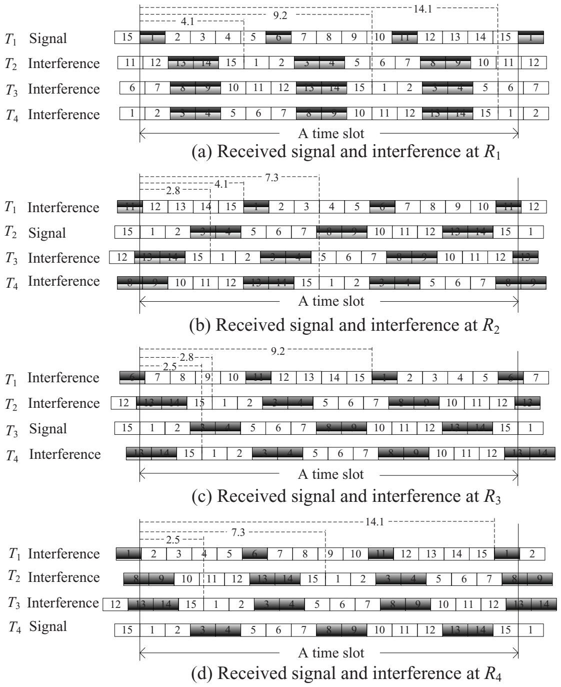

500m
900m
700m
650m
T_1
T_2
T_3
T_4
R_1
R_2
R_3
R_4
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Transmitter T_1: payload symbols in a time slot
Transmitter T_2: payload symbols in a time slot
Transmitter T_3: payload symbols in a time slot
Transmitter T_4: payload symbols in a time slot

(a) A 4-link network (b) A schedule of payloads at transmitters

Figure 1: An example of PD-IA.

Table 1: Propagation delays normalized *with respect to* a symbol duration.

Node	R_1	R_2	R_3	R_4
T_1	3.9	8.0	13.1	18.0
T_2	8.0	3.9	6.7	11.2
T_3	13.1	6.7	3.9	6.4
T_4	18.0	11.2	6.4	3.9



(a) Received signal and interference at R_1

(b) Received signal and interference at R_2

(c) Received signal and interference at R_3

(d) Received signal and interference at R_4

Figure 2: The signal and interference at each receiver.

in Fig. 2(b), (c) and (d), respectively.

Quantitatively, in Fig. 2, we have a total of 21 payload symbols that are successfully transmitted over 15 symbol intervals. That is, this scheduling of payload symbols allows 6 more symbols to be transported over 15 symbol intervals, making an increase of 40% for spectral efficiency.[1] As this example shows, the essence of PD-IA is the design of a scheduling algorithm to exploit the specific propagation delays between the transmitters and the receivers so that at each receiver, the interfering symbols overlap as much as possible while the desired symbols are free of interference.

1.2 Goals of This Paper

Although the idea of PD-IA has been studied by some researchers, the current results are either based on information theory (IT) perspective [3, 6] or limited to physical-layer/single-hop scenario [2, 5]. It remains unclear how PD-IA can be fully exploited in a complex *multi-hop* UWA network. The goal of this paper is to explore PD-IA in UWA networks so that the benefits of PD-IA can be reaped at the network level. Specifically, we are interested in how to take advantage of PD-IA to improve throughput in a multi-hop UWA network with large propagation delays.

1.3 Main Contributions

We propose a TDMA-based frame structure for scheduling and data transmission in a multi-hop UWA network. Under this frame structure, we develop an analytical model for PD-IA in each time slot. Our model consists of a set of constraints such that at each receiver: (i) its *desired* payload symbols are received free of interference, and (ii) the *interfering* payload symbols are allowed to overlap.

Based on this model, we study a throughput maximization problem in a multi-hop UWA network with a set of sessions.

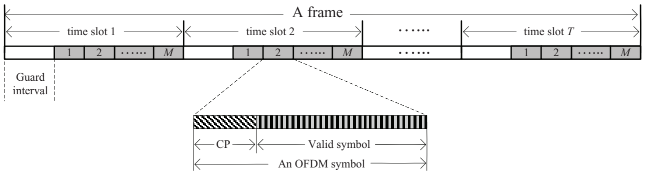


Figure 3: A frame structure.

Specifically, we develop a distributed PD-IA scheduling algorithm, called SHARK-IA, to maximize the minimum rate among a set of sessions. In essence, SHARK-IA is an iterative greedy algorithm that attempts to increase the minimum rate among all active links in each iteration. During each iteration, it chooses a symbol interval for payload so that the interference generated by this new payload symbol is maximally overlapped at its non-intended receivers.

To evaluate the performance of SHARK-IA algorithm, we first compare it to an idealized benchmark algorithm with perfect scheduling and zero propagation delays (similar to the comparison example in Fig. 1). Our simulation results show that SHARK-IA can significantly outperform this idealized benchmark algorithm. Further, we find that the performance gain widens as traffic volume in the network increases. We also compare SHARK-IA algorithm to a centralized solution with perfect PD-IA scheduling. Our simulation results show that SHARK-IA can achieve more than 80% optimal throughput performance by the centralized solution when the number of sessions is small. When the number of sessions becomes large, the centralized solution is no longer computable (even on the supercomputer in our institution), while SHARK-IA can yield a competitive feasible solution very quickly.

1.4 Paper Organization

The remainder of this paper is organized as follows. In Section 2, we develop a PD-IA model for UWA networks. In Section 3, we develop a distributed PD-IA scheduling algorithm. In Section 4, we present simulation results to validate the performance of our algorithm. Section 5 concludes this paper.

2. A PD-IA MODEL

In this section, we first describe the frame structure for scheduling and data transmission. Then, we develop a basic model to study PD-IA in a multi-hop UWA network.

2.1 A Frame Structure

We consider a TDMA frame structure in Fig. 3 for scheduling and data transmission in a network environment [18]. Each node repeats the same frame structure over time. As shown in Fig. 3, a frame is divided into T time slots, each of which consists of a guard interval and M OFDM symbols.

A guard interval is employed at the head of each time slot to eliminate the "tail effect" of the previous time slots from unintended transmitters. That is, the use of guard interval in a time slot allows independent PD-IA scheduling of time slots at all nodes in the network. To serve this purpose, the duration of guard interval should be greater than the maximum propagation delay between any two interfering nodes.

Each OFDM symbol consists of two parts: cyclic prefix (CP) and valid symbol. The part of valid symbol can be used to carry payload packet and the part of CP is used to eliminate the "multipath effect" of channel. We note that our PD-IA scheduling does not require the existence of light-of-

[1] If there were no propagation delays, at most 15 payload symbols can be transported in this network since all links are in the same collision domain.

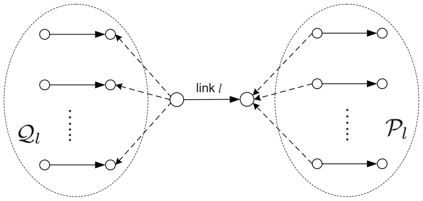


Figure 4: Sets $\mathcal{P}_l$ and $\mathcal{Q}_l$ for link l.

sight channel between any two nodes as [6], since the OFDM modulation can effectively eliminate the inter-symbol interference (ISI) caused by the multipath channel. For example, the OFDM modulation in [12] has a CP of 20 ms duration. Since the delay caused by the multipath channel in UWA networks is typically less than 11 ms [19], this OFDM modulation can completely eliminate the ISI caused by multipath channel.

In this frame structure, half-duplex of a node's transceiver is on time slot level. That is, a node would not change its status (transmitting, receiving, or idling) within a time slot, and can only change its status for a different time slot. In each time slot, the smallest granularity of our PD-IA scheduling is the OFDM symbol. Specifically, for each OFDM symbol, our PD-IA scheduling algorithm is to determine whether or not it is used to carry payload packet.

2.2 Constraints for OFDM Symbol Payload

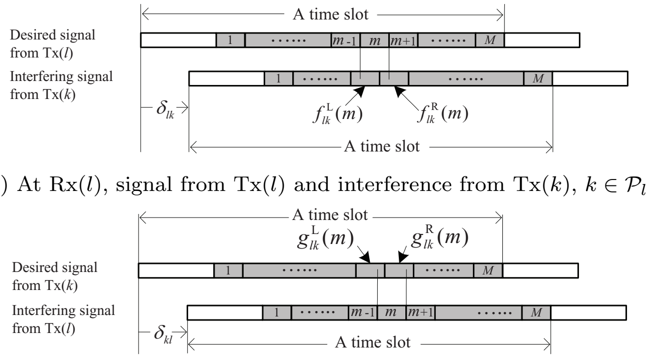
We study the constraints for PD-IA scheduling in each time slot to ensure that at each receiver, its desired payload OFDM symbol is free of interference. Denote $\mathcal{L}$ as the set of links that are traversed by the sessions in the network. Denote $\text{Tx}(l)$ and $\text{Rx}(l)$ as the transmit and receive nodes of link $l \in \mathcal{L}$, respectively. Referring to Fig. 4, denote $\mathcal{P}_l$ as the set of links whose transmitters are interfering with $\text{Rx}(l)$. Similarly, denote $\mathcal{Q}_l$ as the set of links whose receivers are being interfered with by $\text{Tx}(l)$. Now consider $\text{Rx}(l)$. $\text{Rx}(l)$ receives both of the desired signal from $\text{Tx}(l)$ and the interfering signals from $\text{Tx}(k)$, $k \in \mathcal{P}_l$. Suppose that all transmit nodes in the network are synchronized. Then the received signals from intended and unintended transmitters, after taking into consideration their respective propagation delays, will exhibit a time shift with respect to their time slots, as shown in Fig. 5(a). Denote d_{lk} as the Cartesian distance between $\text{Rx}(l)$ and $\text{Tx}(k)$. Denote δ_{lk} as the time offset (in number of OFDM symbols) in a time slot between the desired signal and undesired signal (interference). Then we have

$$\delta_{lk} = \frac{d_{lk} - d_{ll}}{c\tau},$$

where c is the speed of sound in water and τ is the time duration of an OFDM symbol (e.g., $\tau = 85.5$ ms in [12]). Note that a negative value of δ_{lk} indicates that the interfering transmit node $\text{Tx}(k)$ is closer to $\text{Rx}(l)$ than the intended transmit node $\text{Tx}(l)$.

Referring to Fig. 3, denote m as the position of a symbol in a time slot, with $1 \le m \le M$. Denote $z_l(t, m)$ as the indicator of a symbol payload at position m in time slot t for link l. Specifically, $z_l(t, m) = 1$ if the symbol at position m in time slot t is a payload for link l and $z_l(t, m) = 0$ otherwise. For ease of exposition, denote 0 as the position of the guard interval in a time slot. Since a guard interval is filled with null symbols, we have $z_l(t, 0) \equiv 0$ for $l \in \mathcal{L}$ and $1 \le t \le T$.

To explore the constraints for symbol payload, we consider the case for $\text{Rx}(l)$ and $\mathcal{P}_l$ and the case for $\text{Tx}(l)$ and $\mathcal{Q}_l$ separately, as shown in Fig. 5(a) and (b), respectively.



(a) At $\text{Rx}(l)$, signal from $\text{Tx}(l)$ and interference from $\text{Tx}(k)$, $k \in \mathcal{P}_l$.

(b) At $\text{Rx}(k)$, signal from $\text{Tx}(k)$ and interference from $\text{Tx}(l)$, $k \in \mathcal{Q}_l$.

Figure 5: An example to demonstrate the constraints.

Constraints for $\text{Rx}(l)$ **and** $\mathcal{P}_l$**:** As illustrated in Fig. 5(a), for the m-th symbol in a time slot at $\text{Rx}(l)$, it may be interfered with by two consecutive symbols from $\text{Tx}(k)$, $k \in \mathcal{P}_l$. Denote the positions of these two interfering symbols from $\text{Tx}(k)$ as $f_{lk}^{\mathrm{L}}(m)$ and $f_{lk}^{\mathrm{R}}(m)$, respectively. Then we have

$$f_{lk}^{\mathrm{L}}(m) = \begin{cases} m - \lfloor \delta_{lk} \rfloor & \text{if } 1 + \lfloor \delta_{lk} \rfloor \le m \le M + \lfloor \delta_{lk} \rfloor, \\ 0 & \text{otherwise.} \end{cases}$$

$$f_{lk}^{\mathrm{R}}(m) = \begin{cases} m - \lceil \delta_{lk} \rceil & \text{if } 1 + \lceil \delta_{lk} \rceil \le m \le M + \lceil \delta_{lk} \rceil, \\ 0 & \text{otherwise.} \end{cases}$$

Therefore, in time slot t, if the m-th position on link l carries a symbol payload, then the $f_{lk}^{\mathrm{L}}(m)$-th and $f_{lk}^{\mathrm{R}}(m)$-th positions on interfering link k cannot carry a symbol payload. Mathematically, this can be characterized by:

$$z_l(t, m) + \frac{1}{2}\Big[z_k\Big(t, f_{lk}^{\mathrm{L}}(m)\Big) + z_k\Big(t, f_{lk}^{\mathrm{R}}(m)\Big)\Big] \le 1,$$
$$\text{for } k \in \mathcal{P}_l, l \in \mathcal{L}, 1 \le m \le M, 1 \le t \le T. \qquad (1)$$

Constraints for $\text{Tx}(l)$ **and** $\mathcal{Q}_l$**:** As illustrated in Fig. 5(b), at $\text{Rx}(k)$, the m-th symbol in a time slot from $\text{Tx}(l)$ may be interfering with two consecutive symbols from $\text{Tx}(k)$. Denote $g_{lk}^{\mathrm{L}}(m)$ and $g_{lk}^{\mathrm{R}}(m)$ as the positions of these two desired symbols at $\text{Rx}(k)$ that are being interfered with by the m-th symbol from $\text{Tx}(l)$, respectively. Then we have

$$g_{lk}^{\mathrm{L}}(m) = \begin{cases} m + \lfloor \delta_{kl} \rfloor & 1 - \lfloor \delta_{kl} \rfloor \le m \le M - \lfloor \delta_{kl} \rfloor, \\ 0 & \text{otherwise.} \end{cases}$$

$$g_{lk}^{\mathrm{R}}(m) = \begin{cases} m + \lceil \delta_{kl} \rceil & 1 - \lceil \delta_{kl} \rceil \le m \le M - \lceil \delta_{kl} \rceil, \\ 0 & \text{otherwise.} \end{cases}$$

Therefore, in time slot t, if the m-th position from interfering link l carries a symbol payload, then the $g_{lk}^{\mathrm{L}}(m)$-th and $g_{lk}^{\mathrm{R}}(m)$-th positions on link k cannot carry a symbol payload. Mathematically, this can be characterized by:

$$z_l(t, m) + \frac{1}{2}\Big[z_k\Big(t, g_{lk}^{\mathrm{L}}(m)\Big) + z_k\Big(t, g_{lk}^{\mathrm{R}}(m)\Big)\Big] \le 1,$$
$$\text{for } k \in \mathcal{Q}_l, l \in \mathcal{L}, 1 \le m \le M, 1 \le t \le T. \qquad (2)$$

3. SHARK-IA: A DISTRIBUTED PD-IA SCHEDULING ALGORITHM

Consider a multi-hop UWA network with a set of unicast sessions in the network (see Fig. 9 for example). The route from the source node of each session to its destination node is given *a priori,* which can be computed by some distributed

Table 2: The state information for link l.

$s_l(i,t)$	Half-duplex status ("IDLE", "TX", "RX") of link l's transmitter/receiver in time slot t
$z_l(t,m)$	Payload status of the m-th symbol in time slot t for link l
$\mathcal{Q}_l$	The set of links that are interfered with by link l
$\mathcal{P}_l$	The set of links that interfere with link l

routing protocol (e.g., Bellman-Ford algorithm [11]). We develop a distributed payload scheduling algorithm based on the proposed PD-IA model, with the objective of maximizing the minimum rate among the sessions. We first state our assumptions and then explain each key module in the algorithm.

3.1 Assumptions

We have the following assumptions in the design of SHARK-IA.

- Each session has a persistent and latency-tolerant traffic at its source. This assumption helps us to explore the full potential of PD-IA and simplify the discussion of the algorithm.
- The nodes in the network are well synchronized. Some distributed synchronization protocols (see, e.g., [20]) can achieve several microsecond synchronization error within 10 seconds in the UWA environment. Since the duration of OFDM symbol is at the level of 100 ms (e.g., 85.5 ms [12]), our scheduling algorithm is robust to the synchronization error.
- Every node knows the location information of its neighboring nodes. Based on the location information, a node can compute the value of propagation delays between itself and its neighboring nodes. Since the existing distributed localization schemes can achieve the accuracy of 1 m for a 3 km × 4 km area [4], the propagation delay error caused by the inaccurate location information is less than 1 ms. Our scheduling algorithm is also robust to the location information error since it does not require perfect alignment of the OFDM symbols.
- Each node in the network can exchange scheduling information with those nodes inside its interference range. This is a mild assumption since there are many ways to achieve it in a distributed environment. Given that this is not our contribution, we skip its discussion to conserve space.

Since the goal of this paper is to outline a distributed algorithm to show the benefits of PD-IA in a multi-hop network, issues associated with protocol design (e.g., message format, control packet delay and overhead, recovery from lost packet) are beyond the scope of this paper.

3.2 Algorithm Overview

In essence, the proposed distributed algorithm is a greedy algorithm that attempts to increase the minimum rate among all the links traversed by the sessions iteratively. This is equivalent to increasing the rate on each link traversed by each session iteratively (a link traversed by multiple sessions will be considered for multiple times). For a given link, the algorithm attempts to increase its rate by finding a symbol interval that can be used for a payload. Among the eligible symbol intervals, the final choice is determined by IA, in the sense that we wish to have the interference from this symbol to overlap with as many interfering symbols from other links as possible. If we cannot find any such eligible symbol interval in a frame on link l, then we attempt to make some adjustment on the current payload structure in sets $\mathcal{P}_l$ and $\mathcal{Q}_l$ so that some symbol interval can be used for a new payload on link l.

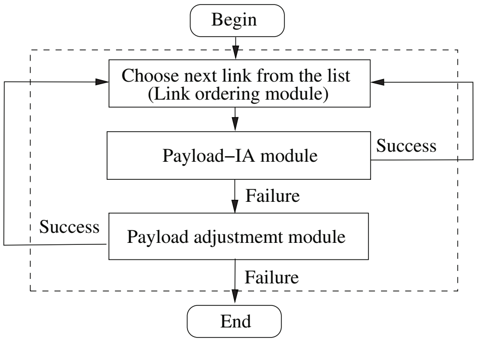


Figure 6: A flow chart of our scheduling algorithm.

The state information that each link l maintains is listed in Table 2. The state information is initialized as follows: $s_l(i,t) =$ "IDLE" for $l \in \mathcal{L}$, $i \in \{\mathrm{Tx}(l), \mathrm{Rx}(l)\}$, and $1 \leq t \leq T$; $z_l(t,m) = 0$ for $l \in \mathcal{L}$, $1 \leq t \leq T$, and $1 \leq m \leq M$.

The flow chart of our scheduling is presented in Fig. 6. As shown in the figure, there are three main modules in the algorithm: *link ordering, payload-IA,* and *payload adjustment*. Next, we explain each of them in detail.

3.3 Link Ordering Module

Our proposed distributed algorithm is a greedy algorithm that attempts to increase the minimum rate among all the links traversed by the sessions iteratively. This is equivalent to increasing the rate of each link traversed by each session iteratively. A straightforward approach is to order all the links traversed by the sessions into a list (a link traversed by multiple sessions will be on the list for multiple times) and consider the links in the list sequentially. Here, we find that the ordering of the links in this list plays an important role in the performance of the algorithm. In our algorithm, we propose to order the links in the list based on their "interference burden", which is defined as follows.

DEFINITION 3.1. The interference burden of a link $l \in \mathcal{L}$, denoted as b_l, is defined as the number of links in $\mathcal{P}_l$ and $\mathcal{Q}_l$, i.e., $b_l = |\mathcal{P}_l| + |\mathcal{Q}_l|$.

By ordering the links in the list based on non-increasing values of interference burden, we are exploiting the most opportunity for interference overlapping from the beginning. The details of how to schedule a symbol payload for a link by exploiting PD-IA will be explained in Sections 3.4 and 3.5. Note that each iteration considers one link in the list. After all the links in the list are considered sequentially, the algorithm returns to the first link in the list in a cyclic manner until the algorithm terminates.

For distributed implementation, we assume that there is a dedicated control channel for scheduling. The status of start or completion of a particular iteration is shared among the nodes via this control channel. To find the order of a link in the link ordering list in a distributed network, we adopt the distributed ranking algorithm by Zaks [21]. Zaks' algorithm was proposed to solve the problem of ranking the nodes in a network with a given initial value in non-decreasing order. To adopt this node ordering algorithm for our link ordering problem, we can have the receiver of link $l \in \mathcal{L}$ maintain the link's interference burden b_l, and then execute the distributed ranking algorithm by treating $1/b_l$ as its initial value (as we are interested in a non-increasing order of links).

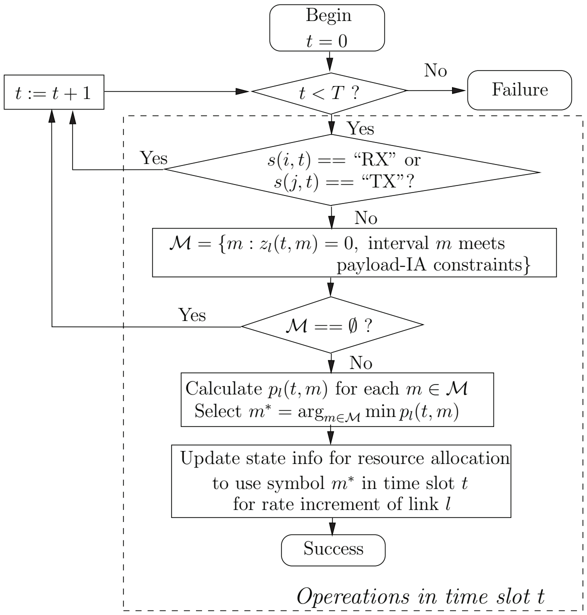

Begin
$t = 0$
$t < T$?
No
Failure
$t := t + 1$
Yes
Yes
$s(i,t)$ == "RX" or
$s(j,t)$ == "TX"?
No
$\mathcal{M} = \{m : z_l(t,m) = 0,$ interval m meets
payload-IA constraints$\}$
Yes
$\mathcal{M} == \emptyset$?
No
Calculate $p_l(t,m)$ for each $m \in \mathcal{M}$
Select $m^* = \arg_{m \in \mathcal{M}} \min p_l(t,m)$
Update state info for resource allocation
to use symbol m^* in time slot t
for rate increment of link l
Success
Opereations in time slot t

Figure 7: A flow chart of payload-IA module.

3.4 Payload-IA Module

The goal of this module is to increase the rate of the current link by one payload symbol without any change of payload on other links. To do so, we consider the time slots in a frame in a sequential order, starting with the first time slot. If the rate increment attempt fails in a time slot, we try the next time slot and so forth, until a rate increment is successful or it fails in all T time slots. A flow chart of payload-IA module is given in Fig. 7.

Choosing a symbol interval in time slot t. Suppose that the current iteration is on link l. Denote i and j as the transmit and receive nodes of link l, i.e., $i = \text{Tx}(l)$ and $j = \text{Rx}(l)$.

At transmit node i, we first check its half-duplex status in time slot t for link l. To consider time slot t for rate increment, its half-duplex status must be "IDLE" or "TX" for link l. Otherwise, this time slot cannot be used for rate increment for link l. Likewise, at receive node j, to consider the same time slot t for rate increment, its half-duplex status must be "IDLE" or "RX" for link l. Otherwise, this time slot still cannot be used for rate increment for link l.

If both transmit node i and receive node j meet the half-duplex requirement, then we move on to find a set of eligible symbol intervals for payload in this time slot. At transmit node i, we identify a set of unused symbol intervals in this time slot that meet the PD-IA constraint (2), which we denote as $\mathcal{M}_i$, i.e.,

$$\mathcal{M}_i = \big\{m : z_l(t,m) = 0, z_k(t, g_{lk}^{\text{L}}(m)) = 0 \text{ and } z_k(t, g_{lk}^{\text{R}}(m)) = 0 \text{ for } k \in \mathcal{Q}_l\big\}.$$

Likewise, at receive node j, we identify a set of unused symbol intervals in this time slot that meet the PD-IA constraint (1), which we denote as $\mathcal{M}_j$, i.e.,

$$\mathcal{M}_j = \big\{m : z_l(t,m) = 0, z_k(t, f_{lk}^{\text{L}}(m)) = 0 \text{ and } z_k(t, f_{lk}^{\text{R}}(m)) = 0 \text{ for } k \in \mathcal{P}_l\big\}.$$

An unused symbol interval is eligible for payload on link l only if this interval is in both $\mathcal{M}_i$ and $\mathcal{M}_j$. Denote $\mathcal{M}$ as the set of such eligible intervals on link l, i.e., $\mathcal{M} = \mathcal{M}_i \cap \mathcal{M}_j$. Although all symbol intervals in $\mathcal{M}$ are eligible for payload, which symbol interval to choose from $\mathcal{M}$ for payload in this iteration is important. Our approach is to choose the one that can create the most interference overlapping shadows on the other links, since this will exploit IA to the fullest extent.

We now consider a link $k \in \mathcal{Q}_l$ and its receive node $\text{Rx}(k)$. For $\text{Rx}(k)$, denote $y_k(t,n)$ as the amount of interference overlapping shadows on its n-th symbol interval in time slot t on link k, which we define as follows:

$$y_k(t,n) = \sum_{h \in \mathcal{P}_k} \Big[z_h\big(t, f_{kh}^{\text{L}}(n)\big) + z_h\big(t, f_{kh}^{\text{R}}(n)\big)\Big], \quad k \in \mathcal{Q}_l, 1 \le t \le T, 1 \le n \le M.$$

As shown in Fig. 4 and Fig. 5(b), symbol interval m from node i (i.e., $\text{Tx}(l)$) is overlapping with the $g_{lk}^{\text{L}}(m)$-th and $g_{lk}^{\text{R}}(m)$-th symbol intervals at $\text{Rx}(k)$, $k \in \mathcal{Q}_l$. If the $g_{lk}^{\text{L}}(m)$-th and $g_{lk}^{\text{R}}(m)$-th symbol intervals at $\text{Rx}(k)$ are already interfered with by other links, then the setting of symbol interval $m \in \mathcal{M}$ for a payload will only align new interference on these already interfered intervals rather than adding interference on some non-interfered intervals. Therefore, we choose a symbol interval that would cast the maximum interference overlapping shadows on the links in $\mathcal{Q}_l$.

Denote $p_l(t,m)$ as the amount of interference overlapping shadows casted by symbol interval $m \in \mathcal{M}$ on the links in $\mathcal{Q}_l$. Then we have

$$p_l(t,m) = \sum_{k \in \mathcal{Q}_l} \Big[1^+\Big(y_k\big(t, g_{lk}^{\text{L}}(m)\big)\Big) + 1^+\Big(y_k\big(t, g_{lk}^{\text{R}}(m)\big)\Big)\Big], \quad (3)$$

where $1^+(x)$ is an indicator function (i.e., $1^+(x) = 1$ if $x \ge 1$ and $1^+(x) = 0$ otherwise).

Denote m^* as the symbol interval that leads to the maximum value of $p_l(t,m)$, $m \in \mathcal{M}$, i.e.,

$$m^* = \arg_{m \in \mathcal{M}} \max p_l(t,m). \quad (4)$$

Then the m^*-th symbol interval in time slot t will be chosen as new symbol payload for rate increment.

Update state information. After choosing m^*-th symbol interval for payload, we update the state information for link l as follows:

- At transmit node i, if $s_l(i,t) =$ "IDLE", then set $s_l(i,t) =$ "TX". Set $z_l(t, m^*) = 1$.
- At receive node j, if $s_l(j,t) =$ "IDLE", then set $s_l(j,t) =$ "RX". Set $z_l(t, m^*) = 1$.

It is easy see that the payload-IA module is amenable to local implementation as all operations of this module are performed at nodes i and j and their neighboring nodes.

3.5 Payload Adjustment Module

As described in Fig. 6, if the payload-IA module fails to increase the rate of the current link l, the payload adjustment module will be invoked. The goal is to increase link l's rate by one symbol payload through adjusting payload structures on links in $\mathcal{P}_l$ and $\mathcal{Q}_l$. In this module, for current link l, we first identify a set $\mathcal{R}$ of symbol intervals over all time slots in a frame on $\text{Tx}(l)$ and $\text{Rx}(l)$ that meet half-duplex constraints but fail to meet the PD-IA constraints. Then we consider a symbol interval in $\mathcal{R}$ iteratively (starting from the one that requires minimum adjustment) and attempt to make some payload adjustments on links in $\mathcal{P}_l$ and $\mathcal{Q}_l$, with the goal of turning the current symbol interval into an eligible interval.

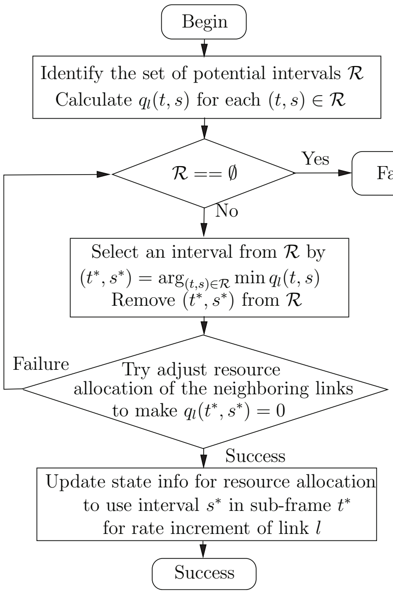

Begin
Identify the set of potential intervals $\mathcal{R}$
Calculate $q_l(t,s)$ for each $(t,s) \in \mathcal{R}$
$\mathcal{R} == \emptyset$
Yes
Failure
No
Select an interval from $\mathcal{R}$ by
$(t^*, s^*) = \arg_{(t,s)\in\mathcal{R}} \min q_l(t,s)$
Remove (t^*, s^*) from $\mathcal{R}$
Failure
Try adjust resource allocation of the neighboring links to make $q_l(t^*, s^*) = 0$
Success
Update state info for resource allocation to use interval s^* in sub-frame t^* for rate increment of link l
Success

Figure 8: A flow chart of payload adjustment module.

The module terminates once we turn an interval in $\mathcal{R}$ into an eligible interval or none of the symbol intervals in $\mathcal{R}$ works out. For the eligible interval, we set it to a payload and update the state information at nodes Tx(l) and Rx(l). A flow chart of the payload adjustment module is given in Fig. 8.

Finding a set of intervals for payload adjustment. Again we denote i and j as the transmit and receive nodes of the current link l, i.e., $i = \text{Tx}(l)$ and $j = \text{Rx}(l)$. For ease of explanation, denote (t, m) as the m-th symbol interval in time slot t.

At transmit node i, we first identify a set of unused symbol intervals over the entire frame that meets half-duplex requirement (as explained in the payload-IA module), which we denote as $\mathcal{R}_i$, i.e.,

$$\mathcal{R}_i = \Big\{(t,m) : z_l(t,m) = 0, s_l(i,t) = \text{“TX” or “IDLE”}\Big\}.$$

Likewise, at receive node j, we identify a set of unused symbol intervals over the entire frame that meets the half-duplex requirement, which we denote as $\mathcal{R}_j$, i.e.,

$$\mathcal{R}_j = \Big\{(t,m) : z_l(t,m) = 0, s_l(j,t) = \text{“RX” or “IDLE”}\Big\}.$$

We consider an unused symbol interval (t, m) and attempt to turn it into an eligible one only if it meets the half-duplex requirement at both transmit node i and receive node j. Denote $\mathcal{R}$ as the set of such unused symbol intervals in a whole frame. Then we have $\mathcal{R} = \mathcal{R}_i \cap \mathcal{R}_j$. Based on the procedure of payload-IA module, we know that the (only) reason why a symbol interval in $\mathcal{R}$ is not eligible for payload is that it fails to meet the PD-IA constraints. To increase the rate of link l by one symbol payload, we attempt to consider a symbol interval in $\mathcal{R}$ one at a time and see if it can be turned into an eligible one by adjusting the current payload structures in sets $\mathcal{P}_l$ and $\mathcal{Q}_l$. Naturally, among the symbol intervals in $\mathcal{R}$, we start with the one that requires the minimum adjustment and so forth.

Suppose that the current unused symbol interval under consideration is (t, m). We now check how much payload adjustment on the links in $\mathcal{P}_l$ and $\mathcal{Q}_l$ is needed if we want to turn it to an eligible interval for payload. For transmit node i, its symbol interval (t, m) is overlapping with symbol intervals $(t, g_{lk}^{\text{L}}(m))$ and $(t, g_{lk}^{\text{R}}(m))$ at Rx(k), $k \in \mathcal{Q}_l$. Should we set (t, m) to a payload at transmit node i, we need to move symbol payloads in intervals $(t, g_{lk}^{\text{L}}(m))$ and $(t, g_{lk}^{\text{R}}(m))$, if they indeed carry payload, to other unused symbol intervals for each link $k \in \mathcal{Q}_l$. Denote q_l^{tx} as the amount of required payload adjustment for setting a payload in interval $(t, m) \in \mathcal{R}$ on link l at Tx(l) (i.e., transmit node i), which we define as:

$$q_l^{\text{tx}}(t,m) = \sum_{k\in\mathcal{Q}_l} \Big[z_k\big(t, g_{lk}^{\text{L}}(m)\big) + z_k\big(t, g_{lk}^{\text{R}}(m)\big)\Big].$$

Likewise, at receive node j, symbol interval (t, m) is overlapping with symbol intervals $(t, f_{lk}^{\text{L}}(m))$ and $(t, f_{lk}^{\text{R}}(m))$ from Tx(k), $k \in \mathcal{P}_l$. Should we set a payload in (t, m) at receive node j, we need to move symbol payload in intervals $(t, f_{lk}^{\text{L}}(m))$ and $(t, f_{lk}^{\text{R}}(m))$, if they indeed carry payload, to other unused symbol intervals for each link $k \in \mathcal{P}_l$. Denote $q_l^{\text{rx}}(t, m)$ as the amount of required payload adjustment for setting a payload in interval $(t, m) \in \mathcal{R}$ on link l at Rx(l) (i.e., receive node j), which we define as:

$$q_l^{\text{rx}}(t,m) = \sum_{k\in\mathcal{P}_l} \Big[z_k\big(t, f_{lk}^{\text{L}}(m)\big) + z_k\big(t, f_{lk}^{\text{R}}(m)\big)\Big].$$

Denote $q_l(t, m)$ as the total amount of required payload adjustment for setting a payload in interval $(t, m) \in \mathcal{R}$ on link l at both Tx(l) and Rx(l). Then $q_l(t, m) = q_l^{\text{tx}}(t, m) + q_l^{\text{rx}}(t, m)$. Among all symbol intervals in $\mathcal{R}$, we choose a symbol interval (t, m) that has the smallest value of $q_l(t, m)$. Denote this symbol interval as (t^*, m^*). We have

$$(t^*, m^*) = \arg_{(t,m)\in\mathcal{R}} \min q_l(t,m).$$

Payload adjustment on links in $\mathcal{P}_l \cup \mathcal{Q}_l$. For interval (t^*, m^*), we try to make necessary payload adjustment on the links in $\mathcal{P}_l \cup \mathcal{Q}_l$ with an attempt to turn this interval to an eligible one. Since link k may be in both $\mathcal{P}_l$ and $\mathcal{Q}_l$, we explain the operations for payload adjustment on each link k by three cases: $k \in \mathcal{Q}_l \backslash \mathcal{P}_l$, $k \in \mathcal{P}_l \backslash \mathcal{Q}_l$, and $k \in \mathcal{P}_l \cap \mathcal{Q}_l$.

Case I: Consider link $k \in \mathcal{Q}_l \backslash \mathcal{P}_l$. At Rx($k$), its symbol intervals $(t^*, g_{lk}^{\text{L}}(m^*))$ and $(t^*, g_{lk}^{\text{R}}(m^*))$ are overlapping with symbol interval (t^*, m^*) from Tx(l). Thus, there are at most two intervals on link $k \in \mathcal{Q}_l \backslash \mathcal{P}_l$ that need adjustment, i.e.,

$$\mathcal{S}_k^{\text{tx}} = \big\{(t^*, n) : z_k(t^*, n) = 1, n \in \{g_{lk}^{\text{L}}(m^*), g_{lk}^{\text{R}}(m^*)\}\big\}.$$

If $\mathcal{S}_k^{\text{tx}} = \emptyset$, then no adjustment is needed. Otherwise, we perform the payload-IA module in Section 3.4 at nodes Tx(k) and Rx(k). If the payload-IA module successfully increases a payload symbol for link k, then we release a payload symbol on link k by setting: $z_k(t^*, n) = 0$ for $(t^*, n) \in \mathcal{S}_k^{\text{tx}}$ at both Tx(k) and Rx(k). We repeat the above operation until both payload symbols in $\mathcal{S}_k^{\text{tx}}$ are released.

Case II: Consider a link $k \in \mathcal{P}_l \backslash \mathcal{Q}_l$. At Rx($l$), its symbol interval (t^*, m^*) are overlapping with symbol intervals $(t^*, f_{lk}^{\text{L}}(m^*))$ and $(t^*, f_{lk}^{\text{R}}(m^*))$ from Tx(k). Thus, there are at most two intervals on link $k \in \mathcal{P}_l \backslash \mathcal{Q}_l$ that need adjustment, i.e.,

$$\mathcal{S}_k^{\text{rx}} = \big\{(t^*, n) : z_k(t^*, n) = 1, n \in \{f_{lk}^{\text{L}}(m^*), f_{lk}^{\text{R}}(m^*)\}\big\}.$$

Again, if $\mathcal{S}_k^{\text{rx}} = \emptyset$, then no adjustment is needed. Otherwise, we follow the same approach in Case I for the payload symbol relocation at nodes Tx(k) and Rx(k).

Case III: Consider link $k \in \mathcal{P}_l \cap \mathcal{Q}_l$. At Rx($k$), its symbol intervals $(t^*, g_{lk}^{\text{L}}(m^*))$ and $(t^*, g_{lk}^{\text{R}}(m^*))$ are overlapping

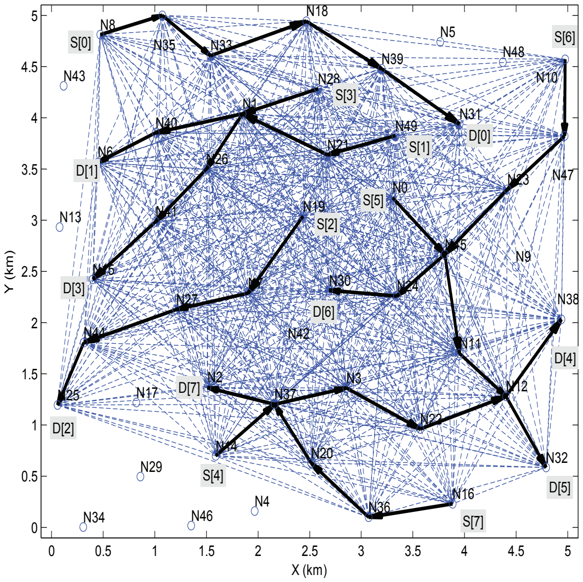


Figure 9: The topology and routing for a network instance.

with symbol interval (t^*, m^*) from $\mathrm{Tx}(l)$; at $\mathrm{Rx}(l)$, its symbol interval (t^*, m^*) are overlapping with symbol intervals $(t^*, f_{lk}^{\mathrm{L}}(m^*))$ and $(t^*, f_{lk}^{\mathrm{R}}(m^*))$ from $\mathrm{Tx}(k)$. Thus, there are at most four intervals on link $k \in \mathcal{P}_l \cap \mathcal{Q}_l$ that need adjustment, i.e.,

$$\mathcal{S}_k^{\mathrm{tx,rx}} = \big\{(t^*, n) : z_k(t^*, n) = 1,$$
$$n \in \{g_{lk}^{\mathrm{L}}(m^*), g_{lk}^{\mathrm{R}}(m^*), f_{lk}^{\mathrm{L}}(m^*), f_{lk}^{\mathrm{R}}(m^*)\}\big\}.$$

Again, if $\mathcal{S}_k^{\mathrm{tx,rx}} = \emptyset$, then no adjustment is needed. Otherwise, we follow the same approach in Case I for the payload symbol relocation at nodes $\mathrm{Tx}(k)$ and $\mathrm{Rx}(k)$.

In the three cases, if any link $k \in \mathcal{P}_l \cup \mathcal{Q}_l$ fails to relocate its payload symbols, we remove symbol interval (t^*, m^*) from set $\mathcal{R}$ and consider the next symbol interval in $\mathcal{R}$, until the link rate is successfully increased or all symbol intervals in $\mathcal{R}$ are removed.

Update state information. If all the links in $\mathcal{P}_l \cup \mathcal{Q}_l$ adjust their payload structure successfully for symbol interval (t^*, m^*) on link l, then we update the state information for link l as follows:

- At transmit node i, if $s_l(i, t^*) =$ "IDLE", then set $s_l(i, t^*) =$ "TX". Set $z_l(t^*, m^*) = 1$.
- At receive node j, if $s_l(j, t^*) =$ "IDLE", then set $s_l(j, t^*) =$ "RX". Set $z_l(t^*, m^*) = 1$.

It is not difficult to see that this module is amenable to local implementation as all operations of this module are restricted on the selected link and its neighboring links.

4. PERFORMANCE EVALUATION

To evaluate the performance of SHARK-IA algorithm, we first compare it to an idealized benchmark algorithm with perfect scheduling and zero propagation delays. We formulate it to an optimization problem and denote it as OPT-noIA. We also compare SHARK-IA algorithm to a centralized algorithm with perfect PD-IA scheduling. We formulate it to another optimization problem and denote it as OPT-IA. The optimal solution to OPT-noIA and OPT-IA may be obtained by using off-the-shelf optimization solver (e.g., IBM CPLEX [9]).

4.1 Simulation setting

We consider networks with 50 nodes being randomly deployed in a 5 km by 5 km area. Among the nodes in the network, there is a set of active sessions with their source and destination nodes being randomly selected among all the nodes. The route from the source node of a session to its destination node is found by the Bellman-Ford algorithm [11].

We assume that all the nodes have the same transmission range 1 km. At a receiving node, we assume that the interference is negligible if the power of the interference is less than -20 dB of the power of its desired signal. Therefore, we set the interference range of a node to 4 km based on the relationship between path loss and distance in underwater acoustic environment [13, Figure 6]. A frame has $T = 10$ time slots, each of which is comprised of a guard interval and $M = 50$ OFDM symbols (see Fig. 3). For each OFDM symbol, we use the same parameters as the "VHF08 EXPERIMENT" in [12], i.e., an OFDM symbol is of 85.5 ms time duration (with a CP of 20 ms time duration). For simplicity, we normalize the time duration of a frame to one unit. We assume that fixed MCS is used for data transmission at payload (OFDM) symbols and each payload (OFDM) symbol carries one data unit.

4.2 A Case Study

Before presenting complete simulation results, we first show results for one network instance as shown in Fig. 9, which has 8 sessions marked by solid arrow line segments in the figure. We apply SHARK-IA algorithm to this network instance. It yields a solution with the objective value of 25 (within unnoticeable amount of time), meaning that the number of payload (OFDM) symbols that can be transported from the source node of each session to its destination node in a frame is 25. We then solve OPT-noIA problem by CPLEX and have an optimal objective value of 18. This implies that SHARK-IA can increase the network throughput by 38.9% when compared to PD-noIA.

4.3 Complete Simulation Results

We consider 7 cases with the above network setting: 1 session, 2 sessions, 3 sessions, 4 sessions, 8 sessions, 12 sessions, and 16 sessions. For each case, we study 100 network instances to obtain their average throughput. Figure 10 exhibits our simulation results, with x-axis being the number of sessions and y-axis being the total throughput of all sessions (i.e., average objective value × the number of sessions). Note that when the number of sessions is greater than 3, the OPT-IA cannot be solved in acceptable amount of time (24 hours per network instance on Blueridge supercomputer at VT).

The simulation results yield the following conclusions: First, SHARK-IA significantly outperforms the OPT-noIA when the number of sessions is greater than two. Second, the throughput gain of SHARK-IA over OPT-noIA increases as the traffic in the network becomes more intensive. Third, when the number of sessions is small (≤ 3), SHARK-IA can achieve more than 80% optimal throughput of the centralized solution (by OPT-IA). Finally, for the network with more than 3 sessions, the optimal centralized solution (OPT-IA) cannot be obtained in reasonable amount of time, while SHARK-IA can yield a competitive solution very quickly.

5. CONCLUSIONS

In this paper, we exploited large propagation delays in UWA networks as an advantage instead of adversary. We developed a PD-IA model that specifies a set of constraints to ensure fea-

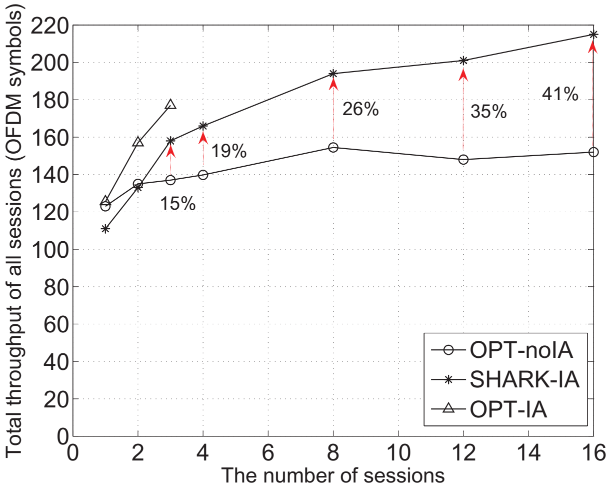


Figure 10: Comparison of SHARK-IA, OPT-noIA, and OPT-IA.

sibility of PD-IA at the physical layer. Based on this model, we studied a network throughput optimization problem with the goal of maximizing the minimum rate among a set of sessions. We developed a distributed PD-IA scheduling algorithm (SHARK-IA) that iteratively increases payloads in a time frame so that at each receiver, (i) the payload symbols from its intended transmitter can be received free of interference, while (ii) the interfering payload symbols from its unintended transmitters can maximally overlap. We validated the performance of SHARK-IA and found significant throughput gains when compared to the case with perfect scheduling and zero propagation delays. More importantly, we found that the throughput gain increases with the traffic intensity in the network, which shows that higher traffic intensity in the network can actually help increase the opportunity to achieve PD-IA.

Acknowledgments

This work was supported in part by the NSF and ONR. The work of Dr. S. Kompella was supported in part by the ONR. Part of Prof. W. Lou's work was completed while she was serving as a Program Director at the NSF. Any opinion, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not reflect the views of the NSF.

6. REFERENCES

[1] Akyildiz, I.F., Pompili, D., and Melodia, T. Underwater acoustic sensor networks: research challenges. *Ad hoc networks,* 3, 3 (2005), 257–279.

[2] Blasco, F.L., Rossetto, F., and Bauch, G. Time interference alignment via delay offset for long delay networks. In *Proc. IEEE GLOBECOM* (Houston, TX, Dec. 2011).

[3] Cadambe, V.R., and Jafar, S.A. Degrees of Freedom of Wireless Networks — What a difference delay makes. In *Proc. IEEE ACSSC* (Pacific Grove, CA, Nov. 2007), 133–137.

[4] Chandrasekhar, V., Seah, W.K., Choo, Y.S., and Ee, H.V. Localization in underwater sensor networks: survey and challenges. In *Proc. ACM international workshop on Underwater networks* (Sep. 2006), 33–40.

[5] Chitre, M., Motani, M., and Shahabudeen, S. Throughput of networks with large propagation delays. *IEEE J. of Ocean. Eng.,* 37, 4 (Oct. 2012) pp. 645–658.

[6] Grokop, L.H., Tse, D.N., and Yates, R.D. Interference alignment for line-of-sight channels. *IEEE Trans. Inf. Theory,* 57, 9 (Sep. 2011), 5820–5839.

[7] Guo, X., Frater, M., and Ryan, M. Design of a propagation-delay- tolerant MAC protocol for underwater acoustic sensor networks. *IEEE J. Ocean. Eng.,* 34, 2 (Apr. 2009), 170–180.

[8] Han, S., Noh, Y., Lee, U., and Gerla, M. M-FAMA: A Multi-session MAC Protocol for Reliable Underwater Acoustic Streams. In *Proc. IEEE INFOCOM* (Turin, Italy, Apr. 2013).

[9] IBM ILOG CPLEX Optimizer, software available at http://www-01.ibm.com/software/integration/optimization/cplex-optimizer.

[10] Kredo, K., Djukic, P., and Mohapatra, P. STUMP: Exploiting position diversity in the staggered TDMA underwater MAC protocol. In *Proc. IEEE INFOCOM* (Rio de Janeiro, Apr. 2009).

[11] Leiserson, C.E., Rivest, R.L., and Stein, C. Introduction to algorithms. (Edited by Cormen, T.H.), The MIT Press, 2001.

[12] Li, B., Huang, J., Zhou, S., Ball, K., Stojanovic, M., Freitag, L., and Willett, P. MIMO-OFDM for high-rate underwater acoustic communications. *IEEE J. of Ocean. Eng.,* 34, 4 (2009), 634–644.

[13] Llor, J., Stojanovic, M., and Malumbres, M.P. A simulation analysis of large scale path loss in an underwater acoustic network. In *Proc. IEEE OCEANS-Spain* (Santander, Spain, June 2011).

[14] Ma, J., and Lou, W. Interference-aware spatio-temporal link scheduling for long delay underwater sensor networks. In *Proc. IEEE SECON* (Salt Lake City, UT, June 2011), 431–439.

[15] Molins, M., and Stojanovic, M. Slotted FAMA: a MAC protocol for underwater acoustic networks. In *Proc. IEEE OCEANS 2006 – Asia Pacific,* (Singapore, May 2007).

[16] Noh, Y., Wang, P., Lee, U., Torres, D., and Gerla, M. DOTS: A propagation delay-aware opportunistic MAC protocol for underwater sensor networks. In *Proc. IEEE ICNP* (Kyoto, Japan, Oct. 2010).

[17] Peleato, B., and Stojanovic, M. Distance aware collision avoidance protocol for ad-hoc underwater acoustic sensor networks. *IEEE Commun. Lett.,* 11, 12 (Dec. 2007), 1025–1027.

[18] Sozer, E., Stojanovic, M., and Proakis, J. Underwater acoustic networks. *IEEE J. of Ocean. Eng.,* 25, 1 (Jan. 2000), 72–83.

[19] Stojanovic, M., and Preisig, J. Underwater acoustic communication channels: Propagation models and statistical characterization. *IEEE Commun. Mag.,* 47, 1 (2009), 84–89.

[20] Syed, A.A., and Heidemann, J.S. Time synchronization for high latency acoustic networks. In *Proc. IEEE INFOCOM* (Barcelona, Spain, Apr. 2006).

[21] Zaks, S. Optimal distributed algorithms for sorting and ranking. *IEEE Trans. Comput.,* 5, 1 (April 1985), 376–379.