

A Distributed Scheduling Algorithm for Underwater Acoustic Networks With Large Propagation Delays

Huacheng Zeng, *Member, IEEE*, Y. Thomas Hou, *Fellow, IEEE*, Yi Shi, *Senior Member, IEEE*,
Wenjing Lou, *Fellow, IEEE*, Sastry Kompella, *Senior Member, IEEE*,
and Scott F. Midkiff, *Senior Member, IEEE*

Abstract—Underwater acoustic (UWA) networks are a key form of communications for human exploration and activities in the oceanographic space of the earth. A fundamental issue of UWA communications is large propagation delays due to water medium, which has posed a grand challenge in UWA network protocol design. Conventional wisdom of addressing this issue is to live with this disadvantage by inserting a guard interval to introduce immunity to propagation delays. Recent advances in interference alignment (IA) open up a new direction to address this issue and promise a great potential to improve network throughput by exploiting large propagation delays. In this paper, we investigate propagation delay-based IA (PD-IA) in multi-hop UWA networks. We first develop a set of simple constraints to characterize PD-IA feasible region at the physical layer. Based on the set of PD-IA constraints, we develop a distributed PD-IA scheduling algorithm to greedily maximize interference overlapping possibilities in a multi-hop UWA network. Simulation results show that the proposed PD-IA algorithm yields higher throughput than an idealized benchmark algorithm without propagation delays, indicating that large propagation delays are not adversarial but beneficial for network throughput performance.

Index Terms—Underwater acoustic networks, interference alignment, distributed scheduling algorithm, large propagation delays.

I. INTRODUCTION

THE RAPID growth of oceanographic data collection, remote sensing, and tactical communications has created a ravenous appetite for acoustic communications in underwater environment [1]. A fundamental issue in underwater acoustic (UWA) communications is large propagation delays that are caused by slow signal (sound) travel speed in water (~ 1500 m/s) [2]. As an example, if the distance between a transmitter and a receiver is 1000 meters, then it takes about 666.7 milliseconds for the signal traveling from the transmitter to the receiver. As expected, such large propagation delays

pose a fundamental challenge in algorithm and protocol design for UWA networks.

To address this issue, research efforts have been spent to alleviate the ill effect of large propagation delays in UWA networks [3]–[9]. Conventional wisdom is to live with this disadvantage by inserting a guard interval to introduce immunity to propagation delays. In other words, these approaches accept the fundamental limitation of large propagation delays and try to develop the best possible protocols and algorithms that live with such limitation. As expected, the throughput under these approaches deteriorates rapidly when the propagation delays become more significant. In this paper, instead of considering large propagation delays as an adversary, we exploit propagation delays as an advantage to improve throughput in multi-hop UWA networks. More specifically, we propose a propagation-delays-based interference alignment (PD-IA) algorithm that exploits propagation delays as an enabling physical phenomenon to project interferences from different transmitters into the same time interval at each receiver. As a result, interferences at each receiver are projected into a squeezed subspace, leaving more time resource available for data transmissions. In what follows, we use a small example to illustrate the basic idea of PD-IA in UWA networks.

A. PD-IA: A Motivating Example

To see how PD-IA works in UWA networks with large propagation delays, let us consider a 4-link UWA network as shown in Fig. 1(a), where the solid arrow line represents data transmission and the dashed arrow line represents interference. The propagation delays between a transmitter and a receiver can be computed based on their distances (given in Fig. 1(a)) and the speed of sound in water (1500 m/s). We assume that at each transmitter, the data is transmitted in consecutive identical time slots. Each time slot consists of 15 orthogonal frequency division multiplexing (OFDM) symbols, each of which is of 85.5 milliseconds time duration [10]. Then the normalized propagation delays (with respect to the OFDM symbol time duration) between transmitters and receivers can be computed, as listed in Table I.

Suppose that we schedule OFDM symbol payload at each transmitter as shown in Fig. 1(b), where the shadowed intervals represent payload while blank intervals represent idle time. Then for each receiver, it receives one desired symbol stream and three interfering symbol streams. Based on their respective

Manuscript received September 1, 2016; revised November 25, 2016; accepted December 26, 2016. Date of publication January 5, 2017; date of current version March 15, 2017. This work was supported in part by NSF under Grants 1642873, 1617634, 1446478, 1443889, 1343222, and ONR Grant N00014-15-1-2926. The associate editor coordinating the review of this paper and approving it for publication was Y. J. (Angela) Zhang.

H. Zeng is with the University of Louisville, Louisville, KY 40292 USA.
Y. T. Hou, Y. Shi, W. Lou, and S. F. Midkiff are with the Virginia Polytechnic Institute and State University, Blacksburg, VA 24061 USA.
S. Kompella is with the U.S. Naval Research Laboratory, Washington, DC 20375 USA.

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TCOMM.2017.2647940

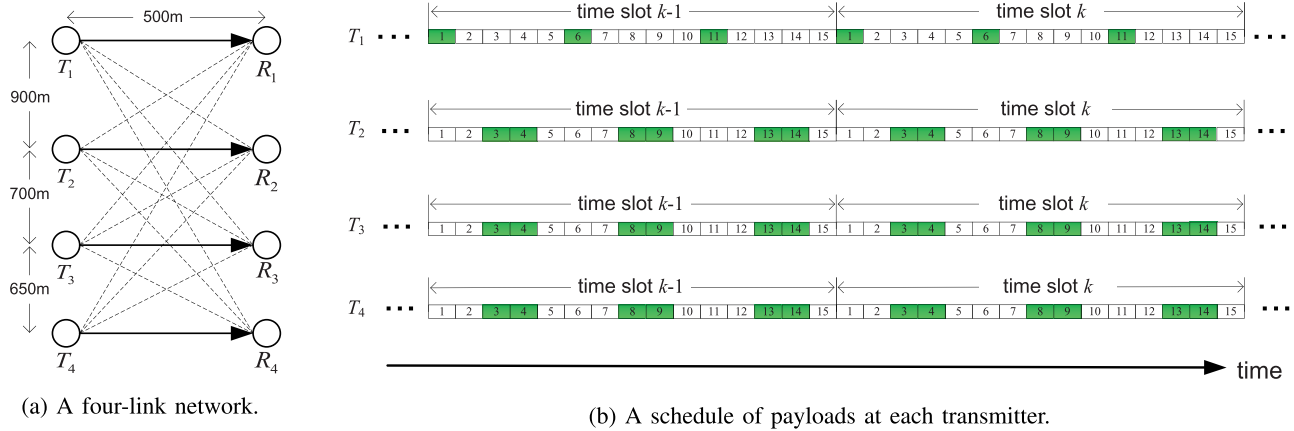


Fig. 1. An example of PD-IA.

TABLE I
PROPAGATION DELAYS NORMALIZED WITH
RESPECT TO A SYMBOL DURATION

Signal path	Delay	Signal path	Delay
$T_1 \rightarrow R_1$	3.9	$T_1 \rightarrow R_2$	8.0
$T_2 \rightarrow R_1$	8.0	$T_2 \rightarrow R_2$	3.8
$T_3 \rightarrow R_1$	13.1	$T_3 \rightarrow R_2$	6.7
$T_4 \rightarrow R_1$	18.0	$T_4 \rightarrow R_2$	11.2
Signal path	Delay	Signal path	Delay
$T_1 \rightarrow R_3$	13.1	$T_1 \rightarrow R_4$	18.0
$T_2 \rightarrow R_3$	6.7	$T_2 \rightarrow R_4$	11.2
$T_3 \rightarrow R_3$	3.9	$T_3 \rightarrow R_4$	6.4
$T_4 \rightarrow R_3$	6.4	$T_4 \rightarrow R_4$	3.9

propagation delays in Table I, the received desired and interfering symbol streams at each receiver are shown in Fig. 2. We can see that, thanks to the propagation delays, the desired symbol stream at each receiver is completely separated from the interfering streams in the temporal domain. For example, at receiver R_1 (see Fig. 2(a)), its desired payload symbols from transmitter T_1 are completely free of interference. On the other hand, its interfering payload symbols (payload symbols from transmitters T_2 , T_3 , and T_4) are overlapping with each other in the temporal domain. Similar separation of desired symbols and alignment of interfering symbols can be observed at receivers R_2 , R_3 , and R_4 .

Quantitatively, we have a total of 21 payload symbols that are successfully transported in a time slot in this network. If there were no propagation delays, at most 15 payload symbols can be transported in a time slot since all links are in the same interference domain. Therefore, by using PD-IA, 6 more payload symbols can be transported over 15 symbol intervals, offering an increase of 40% in spectral efficiency. Note that we use the throughput of the zero-delay case as our performance benchmark because it represents a performance upper bound for the class of algorithms that fight with propagation delays (e.g., [4], [10]). Without PD-IA, such a throughput (15 payload symbols in this network) cannot be achieved in UWA networks

due to the existence of propagation delays. So the throughput of the zero-delay case is an ideal performance benchmark to evaluate the throughput gain of PD-IA in UWA networks. As this example shows, the essence of PD-IA is the design of a scheduling algorithm to exploit the specific propagation delays between transmitters and receivers so that at each receiver, the interfering symbols overlap as much as possible while the desired symbols are free of interference.

B. Goals of This Paper

Although similar idea of PD-IA has been studied by some researchers, the current results are either based on information theory (IT) perspective [11], [12] or limited to single-hop scenarios [13], [14]. It remains unclear how PD-IA can be applied to a multi-hop UWA network. The goal of this paper is to fill this gap by studying PD-IA in multi-hop 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 for a multi-hop UWA network with large propagation delays.

C. 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 simple constraints to ensure 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 unicast sessions. Specifically, we develop a distributed PD-IA scheduling algorithm to maximize the minimum rate among all the sessions in the network. In essence, the PD-IA algorithm is an iterative greedy algorithm that strives to increase the minimum rate among all active links in each iteration. During each iteration, it carefully chooses a symbol interval for payload to make sure that the interference generated by this new payload symbol is maximally overlapped at its non-intended receivers.

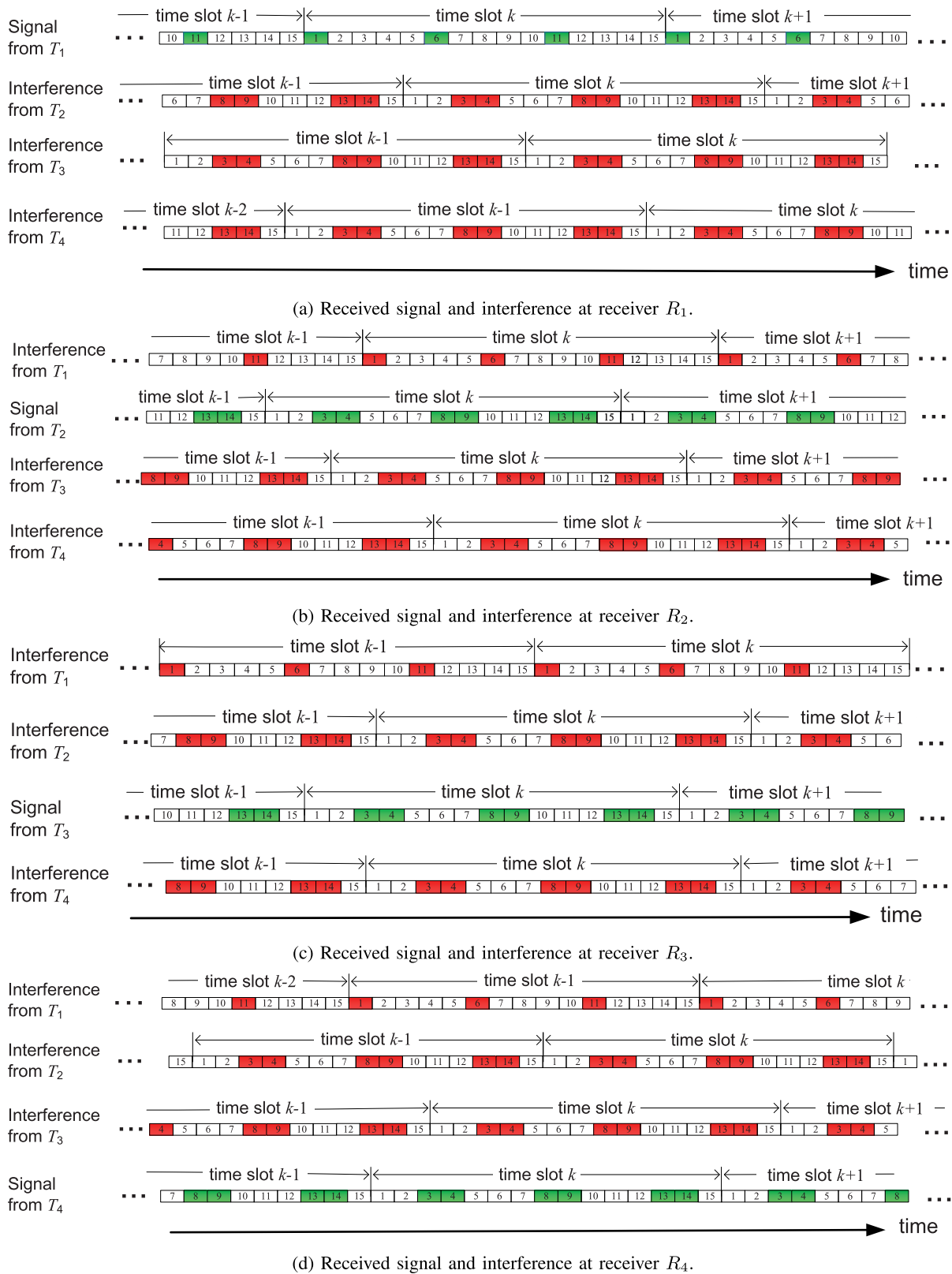


Fig. 2. Received signal and interference at each receiver.

To evaluate the performance of the distributed PD-IA algorithm, we first compare it with an idealized benchmark algorithm with zero propagation delays and perfect scheduling (similar to the comparison in the motivating example

in Fig. 1). Our simulation results show that the distributed PD-IA algorithm can significantly outperform this idealized benchmark algorithm, indicating large propagation delays are not adversarial but beneficial for network throughput

performance. Furthermore, we find that the performance gain becomes more significant as traffic volume in the network increases. We also compare the distributed PD-IA algorithm against a centralized solution with perfect PD-IA scheduling. Our simulation results show that the distributed PD-IA algorithm can achieve more than 80% optimal throughput obtained 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 the distributed PD-IA algorithm can yield a competitive feasible solution very quickly.

D. Paper Organization

The remainder of this paper is organized as follows. In Section II, we present related work on PD-IA. In Section III, we develop a PD-IA model for UWA networks. In Section IV, we develop a distributed scheduling algorithm to leverage the benefits of PD-IA in a multi-hop UWA network. Section V presents our simulation results and Section VI concludes this paper.

II. RELATED WORK

In UWA networks, large propagation delays were considered a fundamental issue that adversely affects the network throughput performance. Research efforts have been spent to alleviate this ill effect in various ways. Molins and Stojanovic [7] proposed an MAC protocol to address the large propagation delays problem in UWA networks. The protocol was based on slotted floor acquisition multiple access (S-FAMA) that combined both carrier sensing and a conversion between the sender and the sinker prior to data transmission. Peleato and Stojanovic [9] developed a distance-aware channel access protocol based on RTS/CTS shaking. By taking into account propagation delays, their developed protocol was free of collision and thus outperformed CS-ALOHA and slotted FAMA. Kredon *et al.* [5], proposed a staggered TDMA-based scheduling algorithm (called STUMP) for UWA networks, with the objective of increasing channel utilization by taking into account propagation delays. They showed that their scheduling algorithm was superior to the conventional TDMA-based scheduling in terms of throughput. Guo *et al.* [3] proposed a RTS/CTS-based protocol for the networks with large propagation delays. With the information of propagation delays, the protocol enabled simultaneous data transmissions among the nodes while avoiding collision. Noh *et al.* [8] proposed the DOTS protocol, in which the node uses neighbors' propagation delay map and the expected transmission schedules to increase the chances of concurrent transmissions while reducing the likelihood of collisions. Han *et al.* [4] proposed a multi-session floor acquisition multiple access (M-FAMA) algorithm based on the RTS/CTS/ACK shaking mechanism. This protocol enabled the senders to initiate multiple concurrent sessions while avoiding collisions by calculating their neighbors' transmission schedules and propagation delays. Although all these efforts [3]–[5], [7]–[9] considered propagation delays in the design of their

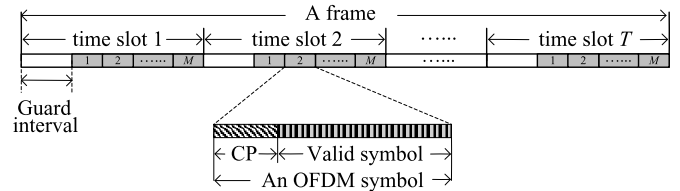


Fig. 3. A frame structure.

algorithms, none of them offered a systematic and disciplined approach to fully exploit propagation delays for throughput maximization.

PD-IA is an effective technique that leverages propagation delays to our advantage for throughput maximization. The concept of IA was coined by Jafar and Shamai for the two-user X channel [15] and its huge potential benefits were substantiated in a seminar paper [16]. Since then, the results of IA have been developed for a variety of channels and networks, such as the K -user MIMO interference channel [17], the X network with arbitrary number of users [18], MIMO-OFDM channel [19], the MIMO Y channel [20], ergodic capacity in fading channel [21], [22], the cellular network [23], and WLAN [24]. As a specific form of IA, PD-IA has been studied in [11], [12], and [14]. Gropop *et al.* [12] studied the PD-IA from the information theoretic perspective. They showed that by using PD-IA, the spectral efficiency of the K -user interference channel can linearly grow with K . However, their result is based on an unpractical assumption of the bandwidth scaling K in order $O(K^{2K^2})$. A similar result was obtained by Cadambe and Jafar [11], based on the assumption of infinite bandwidth. Chitre *et al.* [14] explored the probability to improve the network throughput by exploiting propagation delays. They developed a scheduling algorithm to harvest the benefits of propagation delays. But their results were limited to single-collision domain and single-hop scenario. Although all these efforts exploited propagation delays to improve the network throughput, their results are limited in the PHY layer and single-hop scenario. In this paper, we advance PD-IA a further step by studying the PD-IA in multi-hop networks.

III. A MATHEMATICAL MODEL OF PD-IA

In this section, we develop a mathematical model to study PD-IA. Such a model is important for studying PD-IA in a multi-hop UWA network from a networking perspective. In what follows, we first design a frame structure for scheduling and data transmissions, and then derive a set of simple constraints to characterize PD-IA feasible region at the PHY layer. We list our notation in Table II.

A. A Frame Structure

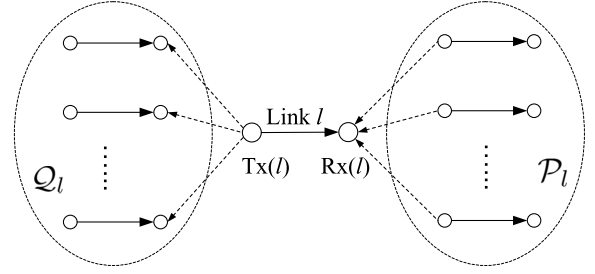
We propose a TDMA-based frame structure, as shown in Fig. 3, for link scheduling and data transmissions in a multi-hop UWA network. Each node repeats the same frame structure over time. As shown in the figure, a frame is divided into T time slots, each of which consists of a guard interval and

TABLE II
NOTATION

Network setting	
\mathcal{F}	The set of sessions in the network
F	The number of sessions in the network
\mathcal{L}	The set of links traversed by any session
L	The number of links in the network
$\mathcal{L}_i^{\text{in}}$	The set of possible incoming links at node i
$\mathcal{L}_i^{\text{out}}$	The set of possible outgoing links at node i
\mathcal{N}^k	The set of nodes in the network
N	The number of nodes in the network
T	The number of time slots in a frame
$\text{Tx}(l)$	The transmitter of link l
$\text{Rx}(l)$	The receiver of link l
$\text{src}(f)$	The source node of session f
$\text{dst}(f)$	The destination node of session f
PD-IA modeling	
M	The number of OFDM symbols in a time slot
\mathcal{P}_l	The set of links whose receiver is within the interference range of $\text{Tx}(l)$
\mathcal{Q}_l	The set of links whose transmitter is within the interference range of $\text{Rx}(l)$
τ	The time duration of an OFDM symbol (e.g., 85.5 ms)
δ_{lk}	The symbol offset of signals from $\text{Tx}(l)$ and $\text{Tx}(k)$ at $\text{Rx}(l)$
d_{lk}	The distance between $\text{Rx}(l)$ and $\text{Tx}(k)$
$f_{lk}^L(m)$	The index of the left symbol on link $k \in \mathcal{P}_l$ that causes collision at the m th symbol on link l
$f_{lk}^R(m)$	The index of the right symbol on link $k \in \mathcal{P}_l$ that causes collision at the m th symbol on link l
$g_{lk}^L(m)$	The index of the left symbol on link $k \in \mathcal{Q}_l$ that is collided by the m th symbol on link l
$g_{lk}^R(m)$	The index of the right symbol on link $k \in \mathcal{Q}_l$ that is collided by the m th symbol on link l
$z_l(t, m)$	The activity of the m th symbol in time slot t on link l (1 if it is used as a payload and 0 otherwise)
Scheduling algorithm design	
\mathcal{M}	A set of eligible symbol intervals in time slot t on link l that are eligible for payload
\mathcal{R}	A set of symbol intervals on link l that can be potentially used for payload
b_l	The potential interference burden of link l
c_l	The average data rate of link l in a time frame
$p_l(t, m)$	The amount of interference overlapping shadows created by symbol (t, m) on link l
$q_l(t, m)$	The amount of payload adjustment for symbol interval (t, m) on link l
$s(i, t)$	The status of node i in time slot t
$y_l(t, m)$	The amount of interference overlapping shadows at symbol (t, m) on link l

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. It allows independent PD-IA scheduling of two consecutive 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 [2]. It is worth pointing out that the use of guard interval in our design differs from that in the prior arts [1]. In our design, a time slot has only one guard interval. But in the prior arts, a time slot has M guard intervals (each OFDM symbol has a guard interval). For example, if $M = 50$, the overhead due to guard interval in our design is only 2% of that in the prior arts.

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 the channel between a transmitter and a receiver. We note that our PD-IA design does

Fig. 4. The set \mathcal{P}_l of links that are interfering with link l and the set \mathcal{Q}_l of links with which link l are interfering.

not require the existence of line-of-sight channel between any two nodes as [12], since the OFDM modulation can effectively eliminate the inter-symbol interference (ISI) caused by the multipath channel. For example, the OFDM modulation in [10] has a CP of 20 ms duration. Since the delay caused by the multipath channel in UWA networks is typically less than 11 ms [25], this OFDM modulation can completely eliminate the ISI caused by multipath channel.

In this frame structure, we assume that a node can switch its role as a transmitter or a receiver at the time slot level. That is, a node would not change its status (transmitting, receiving, or idling) within a time slot; it only changes its status for a different time slot. In each time slot, the smallest granularity of our PD-IA scheduling is OFDM symbol. That is, for each OFDM symbol, our PD-IA design is to determine whether or not it is used to carry payload packet.

B. Constraints for PD-IA Scheduling

Based on the frame structure in Section III-A, we study the constraints for PD-IA scheduling in each time slot. Specifically, we develop the constraints for each OFDM symbol in a time slot to ensure that every payload symbol can be successfully received at its receiver.

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 by $\text{Tx}(l)$. Now let’s consider $\text{Rx}(l)$. $\text{Rx}(l)$ receives its desired signal from $\text{Tx}(l)$ and undesired (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 account 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 [10]).

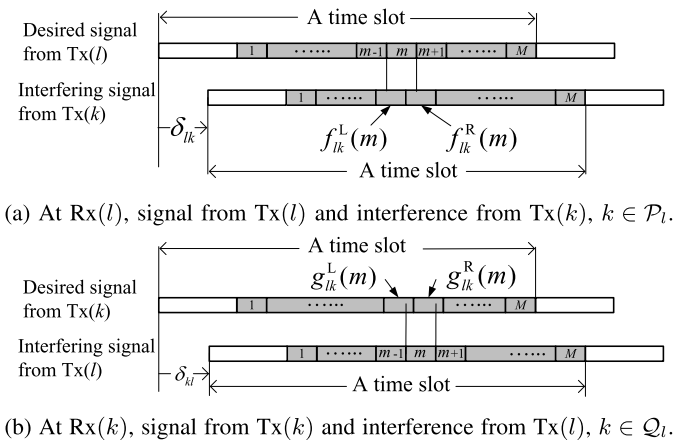


Fig. 5. An example to demonstrate the constraints.

Note that δ_{lk} can be a negative value, which indicates that the interfering transmit node Tx(k) is closer to Rx(l) than the intended transmit node Tx(l).

Referring to Fig. 3, denote m as the position of a symbol in a time slot, with $1 \leq m \leq M$. 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 index of the guard interval in a time slot. Since a guard interval is filled with null symbols, we force $z_l(t, 0) \equiv 0$ for $l \in \mathcal{L}$ and $1 \leq t \leq T$.

Referring to Fig. 4, to derive the constraints for PD-IA scheduling, we first explore the constraints for Rx(l) and \mathcal{P}_l , and then explore the constraints for Tx(l) and \mathcal{Q}_l .

Constraints for Rx(l) and \mathcal{P}_l : For Rx(l) in Fig. 4, it receives desired symbols from Tx(l) and interfering symbols from Tx(k), $k \in \mathcal{P}_l$. Consider the m th desired symbol at Rx(l) as shown in Fig. 5(a). It is interfered by two consecutive undesired symbols from Tx(k), $k \in \mathcal{P}_l$. Denote the positions of these two interfering symbols from Tx(k) as $f_{lk}^L(m)$ and $f_{lk}^R(m)$, respectively.¹ Then we have

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

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

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

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

Constraints for Tx(l) and \mathcal{Q}_l : For Tx(l) in Fig. 4, it sends desired symbols to Rx(l) and interfering symbols to Rx(k), $k \in \mathcal{Q}_l$. Consider the m th desired symbol from Tx(l) as shown

in Fig. 5(b). It is interfering with two consecutive symbols at Rx(k). Denote $g_{lk}^L(m)$ and $g_{lk}^R(m)$ as the positions of these two desired symbols at Rx(k).² Then we have

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

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

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

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

Constraint Summary: Constraints (1) and (2) constitute a mathematical model to ensure that the payload symbols in each time slot in each link can be received free of interference. This model underpins our PD-IA scheduling algorithm design in the next section.

IV. A DISTRIBUTED PD-IA SCHEDULING ALGORITHM

Consider a multi-hop UWA network with a set of unicast sessions in the network (see Fig. 10 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 routing protocol (e.g., the AODV algorithm [26]). 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. To do so, we first state our assumptions and give an overview of the algorithm. Then we detail each key module in the algorithm. Finally, we discuss its complexity.

A. Assumptions

We have the following assumptions in the design of our distributed scheduling algorithm.

- Each session has a persistence 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., [27]) 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 [10]), 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 \times 4 km area [28], the propagation delay

¹In some extreme cases, $f_{lk}^L(m)$ and $f_{lk}^R(m)$ may be equal. Our scheduling algorithm can handle these cases.

²Similarly, in some extreme cases, $g_{lk}^L(m)$ and $g_{lk}^R(m)$ may be equal. Our scheduling algorithm can handle these cases as well.

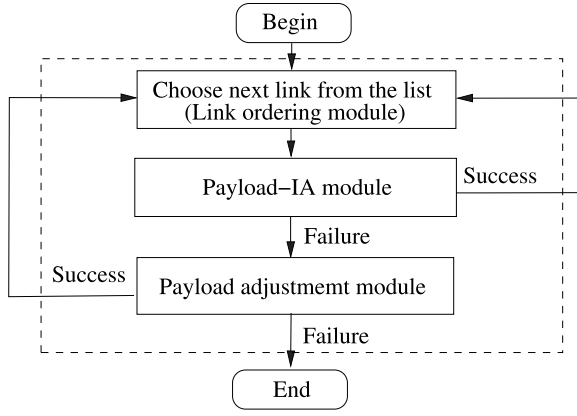


Fig. 6. A flow diagram of our PD-IA scheduling algorithm.

error caused by the inaccurate location information is less than 1 ms. Our scheduling algorithm is also robust to the location information error because 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 UWA 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.

B. Algorithm Overview

In essence, the proposed distributed algorithm is a greedy algorithm that strives 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 try to make adjustment on the current payload structure in sets \mathcal{P}_l and \mathcal{Q}_l so that some symbol intervals can be used for a new payload on link l .

As shown in Fig. 6, there are three main modules in the algorithm: *link ordering*, *payload-IA*, and *payload adjustment*. We briefly describe them as follows:

- *Link Ordering*. The goal of this module is to sort all the links traversed by the sessions into a list. A link traversed by multiple sessions will be on the list for multiple times. For the sorted links in the list, we consider them sequentially and cyclically. To maximize the interference overlapping opportunities from the beginning, we sort the links in the list based on their “interference burden”

values in a non-increasing order. Details are given in Section IV-D.

- *Payload-IA*. The goal of this module is to increase the rate of the selected link by one payload symbol without any payload adjustment on other links. Recall that the time granularity for half-duplex at a node is the time slot. So we find a time slot (starting from the first time slot in a frame) on link l that satisfies half-duplex constraint at both $\text{Tx}(l)$ and $\text{Rx}(l)$. Furthermore, we identify a set of eligible symbol intervals (i.e., the unused symbol intervals that meet the PD-IA constraints) for rate increment in that time slot. If there exist multiple eligible symbol intervals, we choose the one that can produce the most interference overlapping shadows on the neighboring links. Details are given in Section IV-E.
- *Payload Adjustment*. If the payload-IA module fails to find any eligible symbol interval in all time slots, then we attempt to make some necessary adjustment on the current payload structure in \mathcal{P}_l and \mathcal{Q}_l so that some symbol interval on link l can be assigned for a new payload. Specifically, we identify a set \mathcal{R} of symbol intervals (over all time slots in a frame) on link l that meet half-duplex constraints at $\text{Tx}(l)$ and $\text{Rx}(l)$ but fail to meet the PD-IA constraints. Then we iteratively choose a symbol interval in \mathcal{R} (starting from the one that requires the minimum adjustment) and make necessary adjustment in the current payload structure in \mathcal{P}_l and \mathcal{Q}_l , with the aim of turning this symbol interval into an eligible interval. The module terminates once such an interval is found (and thus a symbol payload can be assigned to this interval) or none of the symbol intervals in \mathcal{R} works out. Details are given in Section IV-F.

C. Data Structure

To implement this algorithm in a distributed manner, each node $i \in \mathcal{N}$ in the network needs to maintain the following state information:

- *Frame structure*. Since each node repeats the same transmission/reception/idling behavior at the frame level, it needs to maintain a frame structure as shown in Fig. 3.
- *Incoming/outgoing links*. Each node i maintains the set $\mathcal{L}_i^{\text{in}}$ of its incoming links traversed by the set of unicast sessions in the network. Also, each node i maintains the set $\mathcal{L}_i^{\text{out}}$ of its outgoing links traversed by the set of unicast sessions in the network.
- *Node half-duplex status*. Since time slot is the smallest granularity of half-duplex, each node i needs to maintain its half-duplex status in each time slot for each link. Denote $s_l(i, t)$ as the half-duplex status (“IDLE”, “TX”, “RX”) of node i in time slot t for link $l \in \mathcal{L}_i^{\text{in}} \cup \mathcal{L}_i^{\text{out}}$. $s_l(i, t) = \text{“TX”}$ (“RX”) means that node i is used as the transmitter (receiver) of link l in time slot t .
- *Current payload scheduling*. Each node i needs to maintain the payload scheduling status at symbol level (with or without payload) in each time slot for its associated links. That is, each node i needs to maintain $z_l(t, m)$ for $1 \leq m \leq M$, $1 \leq t \leq T$, and $l \in \mathcal{L}_i^{\text{in}} \cup \mathcal{L}_i^{\text{out}}$.

- *Interfering links.* Referring to Fig. 4, if node i is used as the transmitter of link l , then it needs to maintain the set of links Q_l that is being interfered by link l ; similarly, if node i is used as the receiver of link l , then it needs to maintain the set of links \mathcal{P}_l that are interfering with link l .
- *Node location.* Each node needs to maintain the location information of all the nodes within its interference range. Based on the location information, each node can compute the propagation delays between itself and its neighboring nodes.

The state information at each node i is initialized as follows: $s_l(i, t) = \text{"IDLE"}$ for $i \in \mathcal{N}$, $l \in \mathcal{L}_i^{\text{in}} \cup \mathcal{L}_i^{\text{out}}$, and $1 \leq t \leq T$; $z_l(t, m) = 0$ for $l \in \mathcal{L}$, $1 \leq t \leq T$, and $1 \leq m \leq M$.

D. 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. If a link is traversed by multiple sessions, then it will be considered for multiple times. A straightforward approach is to sort all the links traversed by the sessions into a list and consider the links in the list sequentially and cyclically. 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 sort the links in the list in a non-increasing order based on their interference burden values, which is defined as follows.

Definition 1: For a link $l \in \mathcal{L}$, its interference burden, denoted as b_l , is defined as the number of links in \mathcal{P}_l and Q_l , i.e., $b_l = |\mathcal{P}_l| + |Q_l|$.

By sorting the links in the list based on their values of interference burden in a non-increasing order, we maximize the interference overlapping opportunities from the beginning. The details of how to schedule a symbol payload for a link by exploiting PD-IA will be explained in Sections IV-E and IV-F. 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 [29]. Zaks' algorithm was proposed to solve the problem of ranking the nodes in a network with a given initial value in a non-decreasing order. To adopt this distributed ranking algorithm for our link ordering problem, we can have the receiver of link $l \in \mathcal{L}$ maintain its 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).

E. 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

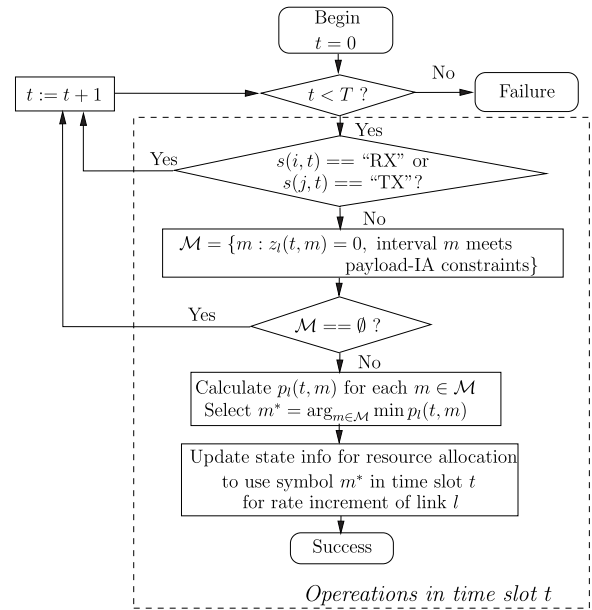


Fig. 7. A flow diagram of the payload-IA module.

other links. To do so, we consider the time slots in a frame in a sequential order, starting from the first time slot. Specifically, we first attempt to increase the rate of the current link in the first time slot; if the rate increment attempt fails in the first time slot, we try the second time slot and so forth, until a rate increment is successful or it fails in all T time slots. The flow diagram 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 $j = \text{Rx}(l)$.

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

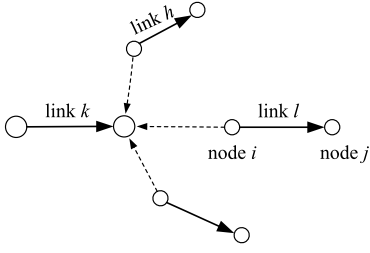
If both transmit node i and receive node j meet the half-duplex requirement in time slot t , then we move on to find a set of eligible symbol intervals for payload in this time slot. At transmit node i , we identify the set of unused symbol intervals that meet the PD-IA constraint (2), which we denote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

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

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

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

An 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 eligible


 Fig. 8. The interference overlapping shadows on link k .

intervals on link l . Then, $\mathcal{M} = \mathcal{M}_i \cap \mathcal{M}_j$. Among the eligible intervals in \mathcal{M} , which symbol interval should be chosen for payload is important. Our algorithm chooses the one that can create the most interference overlapping shadows on the other links, since this is more likely to exploit IA to the fullest extent.

Referring to Fig. 8, we now consider a link that is interfered by link l , say link $k \in Q_l$. At Rx(k), denote $y_k(t, n)$ as the amount of interference overlapping shadows on its n th symbol interval in time slot t , which we define as follows:

$$y_k(t, n) = \sum_{h \in \mathcal{Q}_k} \left[z_h(t, f_{kh}^L(n)) + z_h(t, f_{kh}^R(n)) \right],$$

for $k \in Q_l, 1 \leq t \leq T, 1 \leq n \leq M$.

As shown in Fig. 4 and Fig. 5(b), symbol interval m from node i (i.e., Tx(l)) is overlapping with the $g_{lk}^L(m)$ th and $g_{lk}^R(m)$ th symbol intervals at Rx(k), $k \in Q_l$. If the $g_{lk}^L(m)$ th and $g_{lk}^R(m)$ th symbol intervals at Rx(k) are already interfered by other links (i.e., $y_k(t, g_{lk}^L(m)) \geq 1$ and $y_k(t, g_{lk}^R(m)) \geq 1$), 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 uninterfered intervals. Therefore, we choose a symbol interval that would cast the maximum interference overlapping shadows on the links in 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 Q_l . Then we have

$$p_l(t, m) = \sum_{k \in Q_l} \left[1^+ \left(y_k(t, g_{lk}^L(m)) \right) + 1^+ \left(y_k(t, g_{lk}^R(m)) \right) \right], \quad (3)$$

where $1^+(x)$ is an indicator function (i.e., $1^+(x) = 1$ if $x \geq 1$ 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.,³

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

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

Update State Information: After successfully choosing an unused symbol interval for payload for the current link l , we update the state information at nodes i and j as follows:

³A tie can be handled by any tie-breaking rule, e.g., choosing the symbol interval with the minimum position.

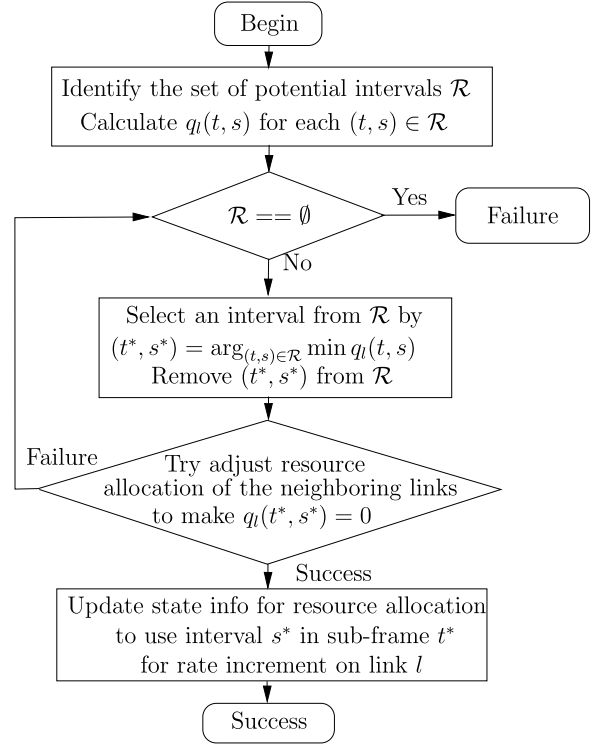


Fig. 9. A flow diagram of the payload adjustment module.

- At transmit node i , if $s_l(i, t) = \text{“IDLE”}$, then set $s_l(i, t) = \text{“TX”}$. Set $z_l(t, m^*) = 1$.
- At receive node j , if $s_l(j, t) = \text{“IDLE”}$, then set $s_l(j, t) = \text{“RX”}$. Set $z_l(t, m^*) = 1$.

It is easy to see that the payload-IA module is amenable to local implementation as all operations of this module (identifying the symbol interval set \mathcal{M} , selecting a symbol interval from \mathcal{M} for payload, and updating state information) are performed at nodes i and j and their neighboring nodes.

F. 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, with the goal of increasing link l 's rate by one symbol payload through adjusting payload structures on the links in \mathcal{P}_l and Q_l .

In this module, for the current link l , we first identify the set of symbol intervals (over all time slots) that meet the half-duplex constraints but fail to meet the PD-IA constraints, which we denote as \mathcal{R} . Then, we consider the symbol intervals in \mathcal{R} sequentially (starting from the one that requires a minimum amount of adjustment) and attempt to make payload adjustment on the links in \mathcal{P}_l and Q_l , with the aim of turning the current ineligible interval into an eligible interval. The module terminates once we successfully turn an interval in \mathcal{R} into an eligible interval or none of the symbol intervals in \mathcal{R} works out. For the resulting eligible interval, we set it to a payload and update the state information at nodes Tx(l) and Rx(l) as well as their neighboring nodes. The flow diagram of the payload adjustment module is given in Fig. 9.

Identify the 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 $j = \text{Rx}(l)$. At transmit node i , we first identify the set of unused symbol intervals (over the entire frame) that meet half-duplex requirement (as explained in the payload-IA module), which we denote as \mathcal{R}_i . Mathematically, we have

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

Likewise, at receive node j , we identify the set of unused symbol intervals (over the entire frame) that meet the half-duplex requirement, which we denote as \mathcal{R}_j . Mathematically, we have

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

For ease of explanation, denote (t, m) as the m th symbol interval in time slot t . We now consider an unused symbol interval (t, m) and attempt to turn it into an eligible one. Symbol interval (t, m) can be turned into an eligible interval 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 all time slots in a frame. We then have $\mathcal{R} = \mathcal{R}_i \cap \mathcal{R}_j$. Based on the procedure of the payload-IA module, we know that the (only) reason why the symbol intervals in \mathcal{R} are not eligible for payload is that they fail to meet the PD-IA constraints. To increase the rate of link l by one symbol payload, we attempt to consider the symbol intervals in \mathcal{R} one at a time and see if it can be turned into an eligible one by adjusting the current payload structures on the links in \mathcal{P}_l and \mathcal{Q}_l . Naturally, among the symbol intervals in \mathcal{R} , we start from the one that requires a minimum amount of 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. At transmit node i , its symbol interval (t, m) is overlapping with symbol intervals $(t, g_{lk}^L(m))$ and $(t, g_{lk}^R(m))$ at $\text{Rx}(k)$, $k \in \mathcal{Q}_l$. Should we set (t, m) to a payload at transmit node i , we need to move symbol payloads in intervals $(t, g_{lk}^L(m))$ and $(t, g_{lk}^R(m))$, 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 $\text{Tx}(l)$. Then we have

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

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

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

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 $\text{Tx}(l)$ and $\text{Rx}(l)$. Then $q_l(t, m) = q_l^{\text{tx}}(t, m) + q_l^{\text{rx}}(t, m)$. Among the symbol intervals in \mathcal{R} , we choose the one (t, m) that has the smallest value of $q_l(t, m)$. Denote the chosen symbol interval as (t^*, m^*) . Then 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 the aim of turning this interval to an eligible one. Depending on link k in \mathcal{P}_l or \mathcal{Q}_l or both, we explain the operations for payload adjustment on link k by three cases: $k \in \mathcal{Q}_l \setminus \mathcal{P}_l$, $k \in \mathcal{P}_l \setminus \mathcal{Q}_l$, and $k \in \mathcal{P}_l \cap \mathcal{Q}_l$.

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

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

If $\mathcal{S}_k^{\text{tx}} = \emptyset$, then no adjustment is needed. Otherwise, we perform the payload-IA module in Section IV-E at nodes $\text{Tx}(k)$ and $\text{Rx}(k)$. 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 $\text{Tx}(k)$ and $\text{Rx}(k)$. We repeat the above operation until both payload symbols in $\mathcal{S}_k^{\text{tx}}$ are released or any of them fails.

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

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

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 adjustment at $\text{Tx}(k)$ and $\text{Rx}(k)$.

Case III: Consider link $k \in \mathcal{P}_l \cap \mathcal{Q}_l$. In this case, at $\text{Rx}(k)$, its symbol intervals $(t^*, g_{lk}^L(m^*))$ and $(t^*, g_{lk}^R(m^*))$ are overlapping with symbol interval (t^*, m^*) from $\text{Tx}(l)$; at $\text{Rx}(l)$, its symbol interval (t^*, m^*) are overlapping with symbol intervals $(t^*, f_{lk}^L(m^*))$ and $(t^*, f_{lk}^R(m^*))$ from $\text{Tx}(k)$. Thus, there are at most four intervals on link k that need adjustment, which we denote as a set as follows:

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

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

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 $\mathcal{P}_l \cup \mathcal{Q}_l$ successfully adjust their payload structure, then symbol interval (t^*, m^*) is eligible for a payload. To increase the rate of link l by assigning a payload at symbol interval (t^*, m^*) , we update the state information at link l 's transmit node i and receive node j as follows:

- At transmit node i , if $s_l(i, t^*) = \text{“IDLE”}$, then set $s_l(i, t^*) = \text{“TX”}$. Set $z_l(t^*, m^*) = 1$.
- At receive node j , if $s_l(j, t^*) = \text{“IDLE”}$, then set $s_l(j, t^*) = \text{“RX”}$. Set $z_l(t^*, m^*) = 1$.

It is easy to see that this module is amenable to local implementation as all operations of this module (identifying the potential payload symbols in \mathcal{R} , payload structure adjustment, and updating state information) are restricted on the selected link and its neighboring links.

G. Computational Complexity and Communication Overhead

Computational Complexity: We analyze the computational complexity of the PD-IA scheduling algorithm at each node as follows. (i) As we explained in Section IV-D, link ordering module can be done in a distributed fashion by employing the distributed ranking algorithm in [29], which has $O(N^3)$ complexity at each node. (ii) Consider the transmit/receive node of the *current* link l . In each iteration, its complexity associated with the payload-IA and payload adjustment modules is $O(T^2M^2)$. Since the maximum number of iterations on link l is $O(TM)$ and there are at most L links, the total complexity at the transmit/receive node of the link l is $O(LT^3M^3)$. (iii) The same transmit/receive node of link l are also involved in the payload-IA and payload adjustment modules when the iteration is working on a link $k \in \mathcal{P}_l \cup \mathcal{Q}_l$. The complexity on the transmit/receive node of link l is $O(LT^3M^3)$. In summary, the total complexity at a (transmit/receive) node is $O(LT^3M^3 + N^3)$.

Communication Overhead: We now analyze the control messages that are required in the proposed algorithm. For the link ordering module, since its core component is the ranking algorithm in [29], the number of its control messages, based on the results in [29], is $O(N^2)$. For the payload-IA module, each of its iterations requires $O(L)$ control messages in the worst case. Since the maximum number of its iterations is $O(TM)$, the total amount of control messages in this module is bounded by $O(LTM)$. For the payload adjustment module, each of its iterations requires $O(L^2)$ control messages in the worst case. Since the maximum number of its iterations is $O(TM)$, the total amount of control messages in this module is bounded by $O(L^2TM)$. In summary, the total amount of control messages in this algorithm is $O(L^2TM + N^2)$.

It is worth pointing out that the analysis of computational complexity and communication overhead is performed for the worst case. In practical networks, we expect that the algorithm has a much lower computational complexity and communication overhead, because the number of iterations in the algorithm is much less than its upper bound due to the existence of interference. Considering the capability of current UWA networks, we do not expect any issues with the proposed algorithm for practical purposes.

V. PERFORMANCE EVALUATION

The proposed algorithm is a heuristic approach that greedily maximizes the overlapping shadow of interference in each of its iterations. A theoretical analysis of its performance limit is prohibitively hard because of its involvement of multiple layers and its local implementation. Given that a nontrivial analytical performance bound of the proposed algorithm is not available, we evaluate its performance via simulation. To evaluate the performance of the PD-IA scheduling algorithm, we first compare it to an idealized benchmark algorithm with perfect scheduling and zero propagation delays. We formulate it as an optimization problem and denote it as OPT-noIA. We also compare the distributed PD-IA scheduling algorithm to a centralized algorithm with perfect PD-IA scheduling. We formulate it as another optimization problem and denote it as OPT-IA. The formulation and explanation of the OPT-noIA and OPT-IA are given in Appendix A and Appendix B, respectively. The optimal solution to OPT-noIA and OPT-IA were obtained using off-the-shelf optimization solver (e.g., IBM CPLEX [31]).

A. Simulation Setting

We consider a set of network instances, each of which has 50 nodes randomly deployed in a 5 km by 5 km area. Among the nodes in each network instance, 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 using AODV algorithm [26].

We assume that all the nodes have the same transmission range 1 km.⁴ 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 [30, Fig. 6]. Referring to Fig. 3, a frame has $T = 10$ time slots, each of which is comprised of a guard interval and $M = 50$ OFDM symbols. For each OFDM symbol, we use the same parameters as the “VHF08 EXPERIMENT” in [10], 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.

B. A Case Study

Before presenting complete simulation results, we first show results for one network instance as shown in Fig. 10. In this figure, a solid arrow line represents a link while a dashed line represents a potential interference. There are 8 sessions in this network: session 0 is from N_8 to N_{31} ; session 1 is from N_{49} to N_6 ; session 2 is from N_{19} to N_{25} ; session 3 is from N_{28} to N_{15} ; session 4 is from N_{14} to N_{38} ; session 5 is from

⁴Data transmission over 1 km is short/medium range communication in underwater acoustic sensor networks [1].

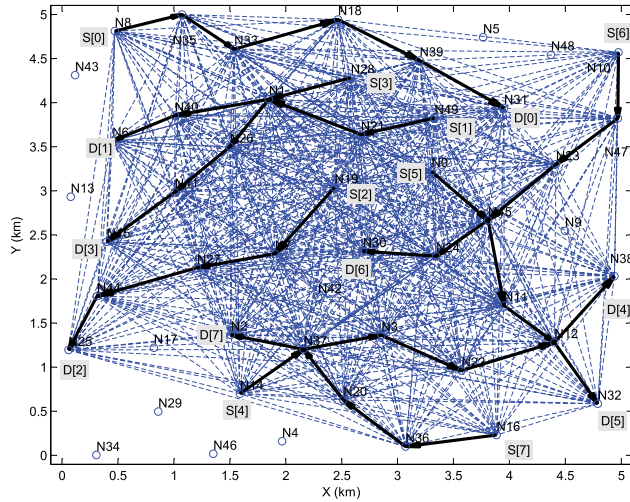


Fig. 10. The topology and routing for a network instance.

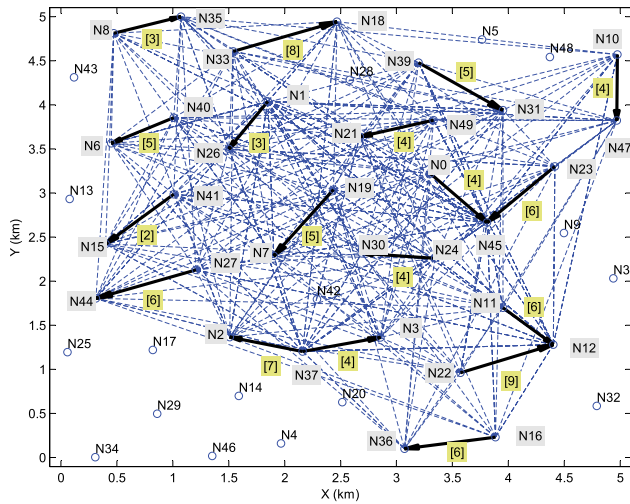


Fig. 11. The scheduling pattern in the first time slot.

N_0 to N_{32} ; session 6 is from N_{10} to N_{30} ; and session 7 is from N_{16} to N_2 .

We apply our proposed distributed PD-IA scheduling algorithm to this network instance. The obtained objective value is 25. We then solve OPT-noIA problem by IBM CPLEX and have an optimal objective value of 18. This implies that our PD-IA distributed scheduling algorithm can increase the network throughput by 38.9% compared to PD-noIA.

We now show some details in the solutions. Fig. 11 shows the payload scheduling results in the first time slot, with the number of payload symbols shown in square brackets next to the active links. Table III summarizes the payload scheduling information in this time slot. In the table, the first column lists the receive node of each active link; the second column lists the number of payload symbols; the third column lists the number of its unused symbol intervals, which is 50 minus the number in the second column; the fourth column lists the number of interfering symbols (from neighboring interfering transmit nodes); and the fifth column lists the interference overlapping ratio, which is the ratio of the fourth column

TABLE III
THE SCHEDULING RESULTS IN THE FIRST TIME SLOT

Active receiver	# of payload symbols	# of unused symbols	# of interfering symbols	Interference overlapping ratio
Rx($N_{19} \rightarrow N_7$)	5	45	86	1.91
Rx($N_8 \rightarrow N_{35}$)	3	47	67	1.42
Rx($N_{33} \rightarrow N_{18}$)	8	42	68	1.62
Rx($N_{39} \rightarrow N_{31}$)	5	45	86	1.91
Rx($N_{49} \rightarrow N_{21}$)	4	46	87	1.89
Rx($N_{40} \rightarrow N_6$)	5	45	67	1.49
Rx($N_{27} \rightarrow N_{44}$)	6	44	75	1.70
Rx($N_1 \rightarrow N_{26}$)	3	47	82	1.74
Rx($N_{41} \rightarrow N_{15}$)	2	48	73	1.52
Rx($N_{37} \rightarrow N_3$)	4	46	84	1.83
Rx($N_{22} \rightarrow N_{12}$)	9	41	66	1.61
Rx($N_0 \rightarrow N_{45}$)	4	46	87	1.89
Rx($N_{11} \rightarrow N_{12}$)	6	44	69	1.57
Rx($N_{10} \rightarrow N_{47}$)	4	46	76	1.65
Rx($N_{23} \rightarrow N_{45}$)	6	44	85	1.93
Rx($N_{24} \rightarrow N_{30}$)	4	46	87	1.89
Rx($N_{16} \rightarrow N_{36}$)	6	44	57	1.30
Rx($N_{37} \rightarrow N_2$)	7	43	80	1.86

to the third column. In the fifth column, a value greater than 1 indicates the existence of interference overlapping. The larger the ratio is, the more PD-IA has been exploited by the algorithm.

Let's take a look at the first row (for receive node Rx($N_{19} \rightarrow N_7$)) in Table III as an example. As shown in Fig. 11, receive node N_7 is within the interference range of all transmit nodes in the network (i.e., $N_0, N_1, N_8, N_{10}, N_{11}, N_{16}, N_{22}, N_{23}, N_{24}, N_{27}, N_{33}, N_{37}, N_{39}, N_{40}, N_{41}$, and N_{49}). The number of interfering payload symbols at receive node N_6 is 86. Since 5 symbol intervals have been used for receiving payloads from intended transmit node N_{19} , there are only 45 unused symbol intervals in which these interfering symbols may fall. That is, the interference overlapping ratio is $86/45 \approx 1.91$ at receiver N_7 .

C. 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 12 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 \times the number of sessions). It should be noted that when the number of sessions is greater than 3, the OPT-IA cannot be solved in a reasonable amount of time (24 hours for a network instance on BlueRidge supercomputer at VT).

The simulation results yield the following conclusions: First, our PD-IA algorithm significantly outperforms the OPT-noIA when the number of sessions is greater than two. Second, the throughput gain of our PD-IA algorithm over OPT-noIA increases as the traffic in the network becomes more intensive. Third, when the number of sessions is small (≤ 3), our PD-IA algorithm can achieve more than 80% optimal throughput of the centralized solution (by OPT-IA). Finally, for the

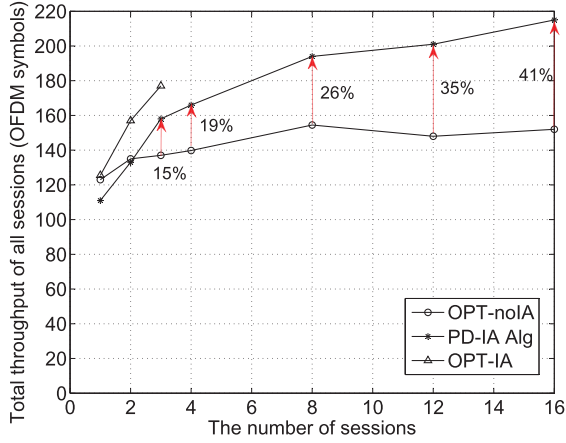


Fig. 12. Comparison of PD-IA algorithm, OPT-noIA, and OPT-IA.

network with more than 3 sessions, the optimal centralized solution (OPT-IA) cannot be obtained in a reasonable amount of time, while our PD-IA algorithm can yield a competitive solution very quickly.

VI. CONCLUSIONS

In this paper, we exploited large propagation delays in multi-hop UWA networks as an advantage instead of adversary. We developed a PD-IA model that specifies a set of constraints to ensure feasibility of PD-IA at the PHY layer. Based on this model, we studied a network throughput optimization problem, with the objective of maximizing the minimum rate among a set of sessions. To solve this optimization problem, we developed a distributed PD-IA scheduling algorithm 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 our PD-IA scheduling algorithm and found significant throughput gains 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.

APPENDIX A

AN IDEALIZED BENCHMARK ALGORITHM WITH PERFECT SCHEDULING AND ZERO PROPAGATION DELAYS

In the literature, research efforts have been spent on the design of underwater MAC protocols with the aim of alleviating the ill effect of large propagation delays [3], [5], [9]. However, the philosophy of these MAC protocols is to fight with large propagation delays (rather than leveraging large propagation delays) and thus their throughput performance is bounded by that under delay-free model. Therefore, to evaluate the throughput performance of our distributed PD-IA scheduling algorithm, we compare it to the throughput performance of the same problem but without using PD-IA under the delay-free model. We denote it as OPT-noIA, which is formulated as follows.

Half-Duplex Constraints: In our scheduling, a node cannot be a transmitter and a receiver in the same time slot. Denote $\alpha_i(t)$ as a binary indicator of a transmitter for node i in time slot t . That is, $\alpha_i(t) = 1$ if node i is a transmitter in time slot t and $\alpha_i(t) = 0$ otherwise. Similarly, denote $\beta_i(t)$ as the indicator of a receiver for node i in time slot t . That is, $\beta_i(t) = 1$ if node i is a receiver in time slot t and $\beta_i(t) = 0$ otherwise. Then we have

$$\alpha_i(t) + \beta_i(t) \leq 1, \quad 1 \leq i \leq N, 1 \leq t \leq T. \quad (5)$$

Link Activity Constraints: Consider a node in a given time slot. If it is a transmitter, then its outgoing links can be active but its incoming links are forced to be inactive; if it is a receiver, then its incoming links can be active but its outgoing links are forced to be inactive; otherwise (being idle), all of its outgoing/incoming links are forced to be inactive. Denote $\mathcal{L}_i^{\text{in}}$ as the set of incoming links to node i and $\mathcal{L}_i^{\text{out}}$ as the set of outgoing links from node i . Denote $x_l(t)$ as the activity of link l in time slot t , i.e., $x_l(t) = 1$ if link l is scheduled to be active in time slot t and $x_l(t) = 0$ otherwise. Then we have

$$\frac{1}{L} \sum_{l \in \mathcal{L}_i^{\text{out}}} x_l(t) \leq \alpha_i(t), \quad 1 \leq i \leq N, 1 \leq t \leq T. \quad (6)$$

$$\frac{1}{L} \sum_{l \in \mathcal{L}_i^{\text{in}}} x_l(t) \leq \beta_i(t), \quad 1 \leq i \leq N, 1 \leq t \leq T. \quad (7)$$

Consider a link in a given time slot. If it is active, then the symbols on this link can be used for payload; otherwise, all of the symbols on this link cannot be used for payload. Thus, we have

$$z_l(t, m) \leq x_l(t), \quad 1 \leq l \leq L, 1 \leq t \leq T, 1 \leq m \leq M. \quad (8)$$

Symbol Activity Constraints: We model the PHY layer behavior of the (OFDM) symbols on each link in each time slot under delay-free model. Consider symbol m on link l in a given time slot. If it is used for payload, then symbol m on link $k \in \mathcal{Q}_l$ cannot be used for payload due to the interference conflict; otherwise, there shall be no constraint on the activity of this symbol. Thus we have

$$z_l(t, m) + \frac{1}{L} \sum_{k \in \mathcal{Q}_l} z_k(t, m) \leq 1, \quad l \in \mathcal{L}, 1 \leq m \leq M, 1 \leq t \leq T. \quad (9)$$

Link Capacity Constraints: For simplicity, we also normalize the time duration of a frame to one unit. We assume that fixed modulation and coding scheme (MCS) is used for data transmission at payload symbols and each payload symbol carries one data unit. Denote c_l as the data rate of link l in a frame. Then we have

$$c_l = \sum_{t=1}^T \sum_{m=1}^M z_l(t, m), \quad l \in \mathcal{L}, 1 \leq t \leq T. \quad (10)$$

For each link l , its aggregate data rate attributed to the sessions cannot exceed its achievable data rate. Thus, we have

$$\sum_{f=1}^F r_l(f) \leq c_l, \quad 1 \leq l \leq L. \quad (11)$$

Flow Balance Constraints: Denote $r(f)$ as the data rate of session f . For the traffic flow from a source to its destination, we denote $r_l(f)$ as the amount of data rate on link l that is attributed to session f . At each node, flow conservation must be observed.

At a source node, we have

$$\sum_{l \in \mathcal{L}_i^{\text{out}}} r_l(f) = r(f), \quad i = \text{src}(f), 1 \leq f \leq F. \quad (12)$$

At an intermediate relay node, we have

$$\sum_{l \in \mathcal{L}_i^{\text{in}}} r_l(f) = \sum_{l \in \mathcal{L}_i^{\text{out}}} r_l(f), \quad 1 \leq i \leq N, i \neq \text{src}(f), \\ i \neq \text{dst}(f), 1 \leq f \leq F. \quad (13)$$

At a destination node, we have

$$\sum_{l \in \mathcal{L}_i^{\text{in}}} r_l(f) = r(f), \quad i = \text{dst}(f), 1 \leq f \leq F. \quad (14)$$

It can be easily verified that if (12) and (13) are satisfied, then (14) is also satisfied. Therefore, it is sufficient to include (12) and (13) in the problem formulation.

Throughput Constraints: Denote r_{\min} as the throughput rate of the bottleneck session. Then we have

$$r_{\min} \leq r(f), \quad 1 \leq f \leq F. \quad (15)$$

By combining the delay-free model with the cross-layer constraints, the throughput optimization problem in a centralized network without using PD-IA technique can be formulated as follows:

OPT-noIA: max r_{\min}
s.t. Half-duplex constraints: (5);
Link activity constraints: (6)–(8);
Symbol activity constraints: (9);
Link capacity constraints: (10)–(11);
Flow balance constraints: (12)–(13);
Throughput constraints: (15).

This formulation is in the form of a mixed integer linear programming (MILP), which is NP-hard in general. However, we find that the OPT-noIA problem for a mid-size network (e.g., a network with 50 nodes) can be solved by CPLEX [31]. We use the optimal objective value of OPT-noIA as the benchmark when evaluating the throughput performance of our distributed PD-IA scheduling algorithm.

APPENDIX B

A CENTRALIZED ALGORITHM WITH PERFECT PD-IA SCHEDULING

The throughput optimization problem in a centralized network with PD-IA, denoted as OPT-IA, can be formulated following the similar procedure as in OPT-noIA. The only difference between OPT-IA and OPT-noIA is the symbol activity constraints. In OPT-IA, the payload scheduling over the OFDM symbols must conform to the PD-IA constraints developed in Section III-B. By replacing (9) with PD-IA constraints (1)–(2), the throughput optimization problem in a

centralized network using PD-IA technique can be formulated as follows:

OPT-IA: max r_{\min}
s.t. Half-duplex constraints: (5);
Link activity constraints: (6)–(8);
PD-IA constraints: (1)–(2);
Link capacity constraints: (10)–(11);
Flow balance constraints: (12)–(13);
Throughput constraints: (15).

This formulation is also in the form of MILP, which is NP-hard in general. Although OPT-IA shares most of constraints with OPT-noIA, it turns out that OPT-IA is much more difficult to solve than OPT-noIA. Our simulation shows that for a 50-node network instance, OPT-IA cannot be solved by IBM CPLEX solver in a reasonable amount of time (24 hours) when the session size is greater than three.

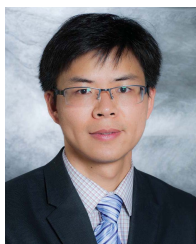
ACKNOWLEDGMENT

Part of 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. The authors thank Virginia Tech Advanced Research Computing for giving them access to the BlueRidge computer cluster.

REFERENCES

- [1] I. F. Akyildiz, D. Pompili, and T. Melodia, "Underwater acoustic sensor networks: Research challenges," *Ad Hoc Netw.*, vol. 3, no. 3, pp. 257–279, Mar. 2005.
- [2] E. M. Sozer, M. Stojanovic, and J. G. Proakis, "Underwater acoustic networks," *IEEE J. Ocean. Eng.*, vol. 25, no. 1, pp. 72–83, Jan. 2000.
- [3] X. Guo, M. R. Frater, and M. J. Ryan, "Design of a propagation-delay-tolerant MAC protocol for underwater acoustic sensor networks," *IEEE J. Ocean. Eng.*, vol. 34, no. 2, pp. 170–180, Apr. 2009.
- [4] S. Han, Y. Noh, U. Lee, and M. Gerla, "M-FAMA: A multi-session MAC protocol for reliable underwater acoustic streams," in *Proc. IEEE INFOCOM*, Turin, Italy, Apr. 2013, pp. 665–673.
- [5] K. Kredto, P. Djukic, and P. Mohapatra, "STUMP: Exploiting position diversity in the staggered TDMA underwater MAC protocol," in *Proc. INFOCOM*, Apr. 2009, pp. 2961–2965.
- [6] J. Ma and W. Lou, "Interference-aware Spatio-temporal link scheduling for long delay underwater sensor networks," in *Proc. IEEE SECON*, Salt Lake City, UT, USA, Jun. 2011, pp. 431–439.
- [7] M. Molins and M. Stojanovic, "Slotted FAMA: A MAC protocol for underwater acoustic networks," in *Proc. OCEANS Asia-Pacific*, May 2007, pp. 1–7.
- [8] Y. Noh, P. Wang, U. Lee, D. Torres, and M. Gerla, "DOTS: A propagation delay-aware opportunistic MAC protocol for underwater sensor networks," in *Proc. ICNP*, Oct. 2010, pp. 183–192.
- [9] B. Peleato and M. Stojanovic, "Distance aware collision avoidance protocol for ad-hoc underwater acoustic sensor networks," *IEEE Commun. Lett.*, vol. 11, no. 12, pp. 1025–1027, Dec. 2007.
- [10] B. Li *et al.*, "MIMO-OFDM for high-rate underwater acoustic communications," *IEEE J. Ocean. Eng.*, vol. 34, no. 4, pp. 634–644, Nov. 2009.
- [11] V. R. Cadambe and S. A. Jafar, "Degrees of freedom of wireless networks—What a difference delay makes," in *Proc. Asilomar Conf. Signals, Syst. Comput. (ACSSC)*, Pacific Grove, CA, USA, Nov. 2007, pp. 133–137.
- [12] L. H. Gropop, D. N. Tse, and R. D. Yates, "Interference alignment for line-of-sight channels," *IEEE Trans. Inf. Theory*, vol. 57, no. 9, pp. 5820–5839, Sep. 2011.
- [13] F. L. Blasco, F. Rossetto, and G. Bauch, "Time interference alignment via delay offset for long delay networks," *IEEE Trans. Commun.*, vol. 62, no. 2, pp. 590–599, Jan. 2014.

- [14] M. Chitre, M. Motani, and S. Shahabudeen, "Throughput of networks with large propagation delays," *IEEE J. Ocean. Eng.*, vol. 37, no. 4, pp. 645–658, Oct. 2012.
- [15] S. A. Jafar and S. Shamai (Shitz), "Degrees of freedom region for the MIMO X channel," *IEEE Trans. Inf. Theory*, vol. 54, no. 1, pp. 151–170, Jan. 2008.
- [16] V. R. Cadambe and S. A. Jafar, "Interference alignment and degrees of freedom of the K -user interference channel," *IEEE Trans. Inf. Theory*, vol. 54, no. 8, pp. 3425–3441, Aug. 2008.
- [17] T. Gou and S. A. Jafar, "Degrees of freedom of the K user $M \times N$ MIMO interference channel," *IEEE Trans. Inf. Theory*, vol. 56, no. 12, pp. 6040–6057, Dec. 2010.
- [18] V. R. Cadambe and S. A. Jafar, "Interference alignment and the degrees of freedom of wireless X networks," *IEEE Trans. Inf. Theory*, vol. 55, no. 9, pp. 3893–3908, Sep. 2009.
- [19] O. El Ayach, S. W. Peters, and R. W. Heath, "The feasibility of interference alignment over measured MIMO-OFDM channels," *IEEE Trans. Veh. Technol.*, vol. 59, no. 9, pp. 4309–4321, Nov. 2010.
- [20] N. Lee, J.-B. Lim, and J. Chun, "Degrees of freedom of the MIMO Y channel: Signal space alignment for network coding," *IEEE Trans. Inf. Theory*, vol. 56, no. 7, pp. 3332–3342, Jul. 2010.
- [21] S. A. Jafar, "The ergodic capacity of phase-fading interference networks," *IEEE Trans. Inf. Theory*, vol. 57, no. 12, pp. 7685–7694, Dec. 2011.
- [22] B. Nazer, M. Gastpar, S. A. Jafar, and S. Vishwanath, "Ergodic interference alignment," in *Proc. IEEE ISIT*, Seoul, South Korea, Jul. 2009, pp. 1769–1773.
- [23] C. Suh, M. Ho, and D. N. C. Tse, "Downlink interference alignment," *IEEE Trans. Commun.*, vol. 59, no. 9, pp. 2616–2626, Sep. 2011.
- [24] S. Gollakota, S. D. Perli, and D. Katabi, "Interference alignment and cancellation," *SIGCOMM Comput. Commun. Rev.*, vol. 39, pp. 159–170, Aug. 2009.
- [25] M. Stojanovic and J. Preisig, "Underwater acoustic communication channels: Propagation models and statistical characterization," *IEEE Commun. Mag.*, vol. 47, no. 1, pp. 84–89, Jan. 2009.
- [26] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (AODV) routing," Internet Engineering Task Force (IETF), Network Working Group (NWG), Tech. Rep. IETF RFC 3561 Jul. 2003.
- [27] A. A. Syed and J. Heidemann, "Time synchronization for high latency acoustic networks," in *Proc. 25th IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Apr. 2006, pp. 1–12.
- [28] V. Chandrasekhar, W. K. G. Seah, Y. S. Choo, and H. V. Ee, "Localization in underwater sensor networks: Survey and challenges," in *Proc. 1st ACM Int. Workshop Underwater Netw. (WUWNet)*, New York, NY, USA, Sep. 2006, pp. 33–40.
- [29] S. Zaks, "Optimal distributed algorithms for sorting and ranking," *IEEE Trans. Comput.*, vol. 5, no. 1, pp. 376–379, Apr. 1985.
- [30] J. Llor, M. Stojanovic, and M. P. Malumbres, "A simulation analysis of large scale path loss in an underwater acoustic network," in *Proc. IEEE OCEANS-Spain*, Santander, Spain, Jun. 2011, pp. 1–5.
- [31] *IBM ILOG CPLEX Optimizer, Software Available at*. accessed on Jan. 2009. [Online]. Available: <https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/>



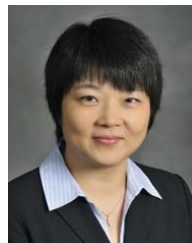
Huacheng Zeng (S'09–M'15) received the Ph.D. degree in computer engineering from Virginia Tech, Blacksburg, VA, in 2015. He is currently an Assistant Professor of Electrical and Computer Engineering with the University of Louisville, Louisville, KY. His research focuses on developing practice solutions to advancing wireless communication systems and enabling innovative wireless applications. He was a recipient of the ACM WUWNET 2014 Best Student Paper Award.



Y. Thomas Hou (F'14) received the Ph.D. degree from the NYU Tandon School of Engineering (formerly Polytechnic University) in 1998. He is currently a Bradley Distinguished Professor of Electrical and Computer Engineering with Virginia Tech, Blacksburg, VA, USA, which he joined in 2002. From 1997 to 2002, he was a member of research staff with the Fujitsu Laboratories of America, Sunnyvale, CA, USA. He has authored over 100 journal papers and 130 conference papers in networking related areas. He holds five U.S. patents. He has authored/co-authored two graduate textbooks: *Applied Optimization Methods for Wireless Networks* (Cambridge University Press, 2014) and *Cognitive Radio Communications and Networks: Principles and Practices* (Academic Press/Elsevier, 2009). His current research focuses on developing innovative solutions to complex science and engineering problems arising from wireless and mobile networks. His papers have been recognized by five best paper awards from the IEEE and two paper awards from the ACM. He has been on the Editorial Boards of a number of the IEEE and the ACM transactions and journals. He is the Steering Committee Chair of the IEEE INFOCOM conference and a member of the IEEE Communications Society Board of Governors. He is also a Distinguished Lecturer of the IEEE Communications Society.



Yi Shi (S'02–M'08–SM'13) is currently a Senior Research Scientist the Intelligent Automation Inc., Rockville, MD, and an Adjunct Assistant Professor with Virginia Tech. He has authored one book, five book chapters, and over 120 papers on wireless network algorithm design and optimization. His research focuses on optimization and algorithm design for wireless networks and social networks. He has co-organized several IEEE and ACM workshops and has been a TPC member of many major IEEE and ACM conferences. He is currently an Editor of the IEEE COMMUNICATIONS SURVEYS AND TUTORIALS. He has been named an IEEE COMMUNICATIONS SURVEYS AND TUTORIALS Exemplary Editor in 2014. He was a recipient of the IEEE INFOCOM 2008 Best Paper Award, the IEEE INFOCOM 2011 Best Paper Award Runner-Up, and the ACM WUWNet 2014 Best Student Paper Award.



Wenjing Lou (F'15) received the Ph.D. degree in electrical and computer engineering from the University of Florida. She is currently a Professor in the Computer Science Department with VA Tech. Her research interests are in the broad area of wireless networks, with special emphases on wireless security and cross-layer network optimization. Since 2014, she has been serving as a Program Director with the National Science Foundation. She is the Steering Committee Chair of the IEEE Conference on Communications and Network Security.



Sastry Kompella (S'04–M'07–SM'12) received the Ph.D. degree in computer engineering from Virginia Tech, Blacksburg, Virginia, in 2006. He is currently the Head of Wireless Network Theory Section, Information Technology Division with the U.S. Naval Research Laboratory, Washington, DC. His research focuses on complex problems in crosslayer optimization and scheduling in wireless and cognitive radio networks.



Scott F. Midkiff (S'82–M'85–SM'92) is currently a Professor and a Vice President of Information Technology and Chief Information Officer with Virginia Tech, Blacksburg, VA. From 2009 to 2012, he was the Head of the Bradley Department of Electrical and Computer Engineering with Virginia Tech. From 2006 to 2009, he served as a Program Director with the National Science Foundation. His research interests include wireless and ad hoc networks, network services for pervasive computing, and cyber-physical systems.