Simplified Scheduling for Underwater Acoustic Networks

Wouter van Kleunen, Nirvana Meratnia, Paul J.M. Havinga
Pervasive Systems, University of Twente
7522 NB Enschede, The Netherlands
Email: {w.a.p.vankleunen, n.meratnia, p.j.m.havinga}@utwente.nl

***Abstract*—The acoustic propagation speed under water poses significant challenges to the design of underwater sensor networks and their medium access control protocols. Similar to the air, scheduling transmissions under water has significant impact on throughput, energy consumption, and reliability.**

In this paper we present an extended set of simplified scheduling constraints which allows easy scheduling of underwater acoustic communication. We also present two algorithms for scheduling communications, i.e. a centralized scheduling approach and a distributed scheduling approach. The centralized approach achieves the highest throughput while the distributed approach aims to minimize the computation and communication overhead. We further show how the centralized scheduling approach can be extended with transmission dependencies to reduce the end-to-end delay of packets.

We evaluate the performance of the centralized and distributed scheduling approaches using simulation. The centralized approach outperforms the distributed approach in terms of throughput, however we also show the distributed approach has significant benefits in terms of communication and computational overhead required to setup the schedule.

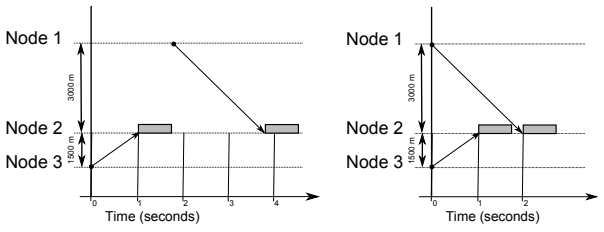
We propose a novel way of estimating the performance of scheduling approaches using the ratio of modulation time and propagation delay. We show the performance is largely dictated by this ratio, although the number of links to be scheduled also has a minor impact on the performance.

I. Introduction

Acoustic communication is the most widely used type of communication for underwater networks. This is because acoustic communication is the only form of communication which allows long-range communication in underwater environments. Acoustic communication, however, poses its own set of challenges for the design of networking and communication protocols. The slow acoustic propagation speed of about 1500 m/s, limited available bandwidth, high transmission energy costs and variations in channel propagation are some of the challenges to overcome.

Examples of existing underwater MAC protocols include T-Lohi [1], Slotted-FAMA [2], and ST-MAC [3]. Scheduled communication approaches, such as ST-MAC [3] and STUMP [4] these have significant benefits over unscheduled approaches such as ST-MAC [3] or ALOHA [5] [6]. These benefits include improved success rate due to the avoidance of packet collision, reduced energy-consumption and improved throughtput. All scheduled based approaches use estimation of the propagation delay to schedule the reception of the packet.

Because of the slow propagation speed and the resulting large propagation times of the signal an uncertainty of the global state of the channel exists, this is called the space-time uncertainty [1]. Because of the spatial-temporal uncertainty, exclusive access to the medium is not required for collision free communication. Rather transmission times should be scheduled such that no collision occurs at reception. Figure 1 shows how two packets can be transmitted at the same time but are received without collision at the receiver. By exploiting the fact that we can have an estimation of the propagation delay, several transmissions can be scheduled at the same time as long as the reception of the packet is scheduled without interference.



(a) Exclusive access (b) Scheduled

Fig. 1. Exploiting spatial-temporal uncertainty in underwater communication with scheduling

Scheduling approaches such as ST-MAC [3] and STUMP [4] are able to exploit this spatial-temporal uncertainty but do so at the cost of complex scheduling algorithms. In [7] we have shown how to derive a simplified set of constraints which greatly simplifies the scheduling of underwater communication. In [8] we have shown how this set of constraints can be used to schedule large-scale networks and in [9] we have evaluated the performance of scheduled communication for an underwater localization system.

In this paper we will review the set of simplified constraints and present a centralized and distributed scheduling approach. We will also extend the centralized scheduling approach with scheduling dependencies which allows enforcing a certain order of transmission. We show that these scheduling constraints can be used to reduce the

This work is supported by the SeaSTAR project funded by the Dutch Technology Foundation (STW).

doi: 10.4304/jnw.8.1.4-14

end-to-end delay of packets in a multihop data collection network by delay the transmission of a parent node until all the packets of children nodes are received.

For performance evaluation, we compare the centralized and distributed scheduling approaches using simulation. We will show that the distributed scheduling approachehas significant benefits in terms of communication and computational overhead, while the centralized approach is able to achieve the highest throughput. We also show that scheduling constraints can be used to reduce the end-to-end delay of packets. We present a novel way to estimate the performance of scheduling approaches using the ratio of modulation time and scheduling time. Finally we evaluate broadcast scheduling and compare its performance with ALOHA.

II. Related work

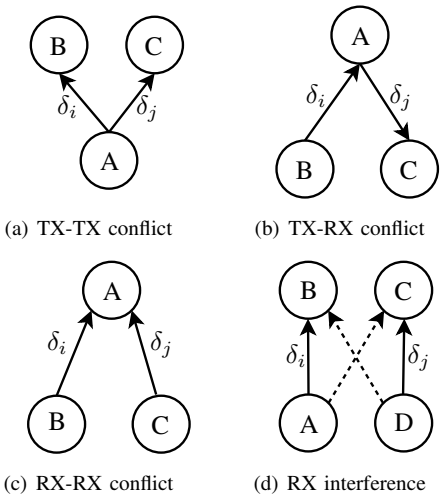
Scheduling communication in underwater communication is done through scheduling the reception of a packet in such a way that it is received without interference from other transmissions active within the network. To do so, the scheduling algorithm needs to know all transmissions and all nodes within the network beforehand and should be able to make an estimation of the propagation delay of the acoustic signal between two nodes. The propagation delay can be estimated by calculating the distance between two nodes using the position information. This distance divided by the propagation speed of the signal ($1500m/s$) results in the propagation delay. Another approach would be to measure the propagation delay at runtime.

Because the propagation delay needs to be estimated and all transmissions should be known before scheduling the transmissions, scheduled communication is most suited for static networks. This is also because the benefits of using a schedule should outweigh the overhead of setting up a schedule. This can usually been done only when the schedule stays valid for a long period of time.

Scheduling algorithms schedule the transmission in time and therefore some form of time-synchronisation is required. This can be done by using a very accurate clock on the nodes or using some form of dynamic time-synchronisation. Time-synchronisation and position estimation have been researched extensively. An example of a time-synchronisation protocol is TSHL [10] and an overview of localization approach can be found in [11].

Scheduling communication can be done using different approaches. For example the scheduling can be done using a slotted approach, such as used by ST-MAC [3] and STUMP [4] or using unslotted approach as employed in our Simplified Scheduling approach [7]. Another difference between scheduling approaches is that they are done using a centralized approach (e.g. ST-MAC [3] and Simplified Scheduling [7]), using a distributed approach (e.g. STUMP [4]) and our Simplified Scheduling approach for large-scale networks [8].

All scheduling algorithms use similar scheduling constraints to model the possible conflicts that may arise. Figure 2 shows these possible conflicts. Our approach to scheduling communication is different from existing approaches because we have simplified these scheduling constraints into a simplified set of scheduling constraints. This allows development of considerably simpler scheduling algorithms.



(a) TX-TX conflict (b) TX-RX conflict

(c) RX-RX conflict (d) RX interference

Fig. 2. Illustration of all possible scheduling conflicts

III. The set of simplified scheduling constraints

In this section we will review the set of simplified scheduling constraints. We will simplify this set of constraints further than what was described in [7] and we will also add support for scheduling broadcasts and scheduling in large-scale networks.

The scheduling constraints place restrictions on the transmissions start time of the "to be scheduled" transmissions. We denote the transmissions tasks as δ where a single transmission task i from the complete set of transmission tasks is denoted as δ_i. For each transmission we need to calculate the transmission start time $\delta_i.start$. Every transmission has a certain duration $\delta_i.duration$, source $\delta_i.src$ and destination $\delta_i.dst$. We assume the function T will give the transmission delay between two nodes. This function can be implemented by calculating the distance between two nodes and using the estimated propagation speed to calculate the propagation delay.

The set of simplified scheduling constraints we have derived in [7] has been shown in Figure 3. We can simplify this set of constraints further by making the following observation: When the source of transmission i is the same as the source of transmission j then the equation (1) can be rewritten as equation (2).

$$T(\delta_i.src, \delta_i.dst) - T(\delta_j.src, \delta_i.dst), \tag{1}$$

$$T(\delta_i.src, \delta_i.dst) - T(\delta_i.src, \delta_i.dst), \tag{2}$$

Equation 2 will always evaluate to 0. The same can be done for the second equation of the maximum from the

$$\text{given } j \text{ for all } i < j, \begin{cases} \delta_j.start \geq \delta_i.start + \delta_i.duration & \text{if } \delta_i.src = \delta_j.src \\ \delta_j.start \geq \delta_i.start + \delta_i.duration + max(& \\ \quad T(\delta_i.src, \delta_i.dst) - T(\delta_j.src, \delta_i.dst), & \text{if } \delta_i.src \neq \delta_j.src \\ \quad T(\delta_i.src, \delta_j.dst) - T(\delta_j.src, \delta_j.dst)) & \end{cases}$$

Fig. 3. Set of simplified scheduling constraints

second rule from the simplified set of constraints. This makes the maximum term of the second rule to be the $max(0, 0)$ when $\delta_i.src = \delta_j.src$ and makes rule one not required.

In [8] we have extended the set of simplified scheduling constraints with an interference condition. This allows scheduling of large-scale networks where nodes may be outside of each others interference range. Two nodes are outside of interference range of each other if the signal of one node results in a received signal strength on the other node which is below a certain threshold (TH_{cp}). The value of this threshold (TH_{cp}) should be chosen in such a way that interfering signals are always below the receiver sensitivity of the node or the interfering signal can be guaranteed to be captured by the transmission.

The received signal strength is dependant on the output power of the sender and the attenuation between the sender and the receiver. The attenuation between nodes depends on the absorption rate of the water and the spreading of the signal. This path loss equation [12] can be written as follows:

$$10 \log(d, f) = k \cdot 10 \log d + d \cdot 10 \log a(f) \qquad (3)$$

The path loss depends on the carrier frequency (f) of the signal as well as the distance (d) between sender, and receiver. The spreading factor is constant, which can either be spherical ($k = 2$), cylindrical ($k = 1$), or something in between. The frequency dependant attenuation is given by the function $a(f)$.

Using this formula we can calculate whether two nodes interfer with each other. Consider two transmissions δ_i and δ_j, which both have source ($\delta_i.src$ and $\delta_j.src$) and destination ($\delta_i.dst$ and $\delta_j.dst$). We will use the path loss function (PL) to calculate the difference of the received signal strengths at the destination of transmissions (δ_j):

$$Interfer(\delta_i, \delta_j) = TRUE \text{ if } PL(\delta_j.src, \delta_j.dst) - PL(\delta_i.src, \delta_j.dst) \leq TH_{cp} \qquad (4)$$

Function (4) will return $false$ if transmission δ_i does not cause interference for transmissions δ_j. We will now show how this equation can be applied to the set of simplified scheduling rules. The interference rule only applies when two nodes are able to interfer with each others transmissions. If $\delta_i.src$ is out of range of $\delta_j.dst$ and if $\delta_j.src$ is out of range of $\delta_i.dst$, there is no interference and therefore no constraint between the two transmissions.

Finally we will review how scheduling of broadcast messages can be added to the set of simplified scheduling constraints. A broadcast message should be scheduled in such a way that on all positions within the network the message can be correctly received. In other words, no collision should occur at any position in the network.

This can be done as follows: when node A broadcasts its message, node B will have to wait until the message of node A passes. Node B can then starts its transmission. When node B transmits immediately after the message from node A has passed node B, the propagation circle of the message from B will always stay within the propagation circle of node A. This means that on any position within the network both messages can be received without any interference. An example of this is shown in Figure 4.

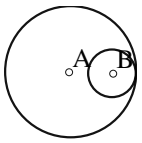


Fig. 4. An example of how two broadcasts can be transmitted without collisions

To put this in a scheduling constraint, node B will have to delay its transmission until the message from A has propagated to the position of B. Assuming the position of A (denoted as $\delta_i.src$) is the source of transmission δ_i, and position of B (denoted as $\delta_j.src$) is the source of transmission δ_j and assuming that we can calculate the propagation time between two positions using the unspecified function T (For example T calculates the Euclidean distance between the two positions divided by an estimation of the sound speed under water), the minimum delay between transmission δ_i and δ_j can now be calculated as:

$$\delta_i.duration + T(\delta_i.src, \delta_j.src)$$

The complete set of simplified scheduling constraints, given in Figure 5, consists now of the following three scheduling rules:

1) If any of the two transmissions is a broadcast, the broadcast scheduling constraint should be used.
2) If both transmissions are unicasts, the interference rule from [7] should be used. This rule ensures that the second unicast arrives at the receiver when the first unicast has been received completely.

3) If both transmissions are unrelated, both can be scheduled at the same time.

IV. SCHEDULING ALGORITHMS

The extended set of simplified constraints, described in Section III, can be applied to design a scheduling algorithm with low complexity. In this section we present the following three algorithms:

- A centralized scheduling approach with high throughput. This approach assumes all information is collected at a central node *prior* to scheduling. Because all information is available at a single point this approach will achieve the *highest throughput*.
- A distributed scheduling approach with *low computational and communication complexity* for large-scale underwater networks. This approach provides a trade-off between the efficiency of the resulting schedule and the amount of communication and computation required to setup the schedule.
- A centralized scheduling approach with transmission dependencies for *low-latency end-to-end communication*. While the first centralized scheduling approach optimizes for througput, this approach shows how transmission dependencies can be used for optimizing the end-to-end delay.

The centralized scheduling algorithm is a reduced complexity version of the algorithm shown in [7]. The algorithm tries to determine a schedule with minimal time length and therefore aims to achieve the highest throughput possible. The distributed algorithm uses the first algorithm to schedule clusters and allows scheduling of large-scale networks with reduced complexity. Finally the last algorithm is a centralized scheduling approach that uses dependencies between transmissions to optimize the end-to-end delay over multihop communication. Rather than trying to achieve the highest throughput possible, this approach minimizes the average time it takes for all packets to travel over multiple hops.

We have chosen these three approaches because we believe they can be used in existing and future underwater sensor network applications. The first approach provides a simple approach for scheduling small-scale networks. The second approach shows how scheduling can be done in large-scale networks and provides a balance between network setup overhead and throughput. The last approach can be used to reduce the end-to-end delay of packets or for applications requiring certain transmissions order such as data-aggregation and other distributed processing approaches.

A. A centralized scheduling approach with high throughput

The extended set of simplified constraints can be applied to design a scheduling algorithm with low complexity for underwater networks. The algorithm from [7], which has $O(n^3)$ complexity, considers every transmission as the first transmission. To reduce the complexity, we can take the first transmission as the transmission to be scheduled at time 0. This will reduce the complexity of the algorithm from $O(n^3)$ to $O(n^2)$.

$V \leftarrow transmissions$ {Set of all transmissions}
$schedule \leftarrow [N] = 0$ {Resulting schedule}
$schedule[0] = 0$ {Schedule the first transmission}
$time = 0$
$last = 0$
$V \leftarrow V \setminus \delta_0$ {Remove transmission from set}
{Scheduling loop schedules transmissions greedy}
while $!empty(V)$ **do**
  $time_{min} \leftarrow infinity$
  {Calculate minimum starting time for remaining transmissions}
  for $\delta \in V$ **do**
    $schedule[\delta_{index}] = max(schedule[\delta_{index}], time + constraint(\delta_{last}, \delta_{index})]$
    {See if this transmission has the smallest starting time}
    if $schedule[\delta_{index}] < time_{min}$ **then**
      $time_{min} \leftarrow schedule[\delta_{index}]$
      $index \leftarrow \delta_{index}$
    end if
  end for
  {Schedule transmission with smallest starting time first}
  $time = time_{min}$
  $last = index$
  $V \leftarrow V \setminus \delta_{index}$
end while

Fig. 6. Reduced complexity algorithm for scheduling transmissions.

The algorithm initially schedules the first transmission. Inside the scheduling loop first all the minimum starting times for the remaining transmissions are calculated. The loop also finds the transmission with the minimum schedule time and removes this transmission from the set of "to be scheduled" transmissions. This is repeated until all transmissions are scheduled.

The algorithm continously updates the start time for the unscheduled transmissions. This is done by calculating the maximum of the previously calculated start time and the new start time calculated using the scheduling constraints. It ensures collision free reception by taking the maximum start transmission time.

When we calculate the schedule only once, there is also no need anymore to precalculate a table of delays for all transmission pairs. Any transmission pair will be considered at most once, but some will never be calculated. At the first iteration the algorithm will calculate the delays for $n-1$ pairs, in the second iteration for $n-2$, and so forth. This will further reduce the complexity from $O(n^2)$ to $O(\frac{1}{2}n^2)$. Because we do not calculate the delay table, the memory space complexity can also be reduced to $O(n)$.

The full algorithm can be seen in Figure 6.

B. A distributed scheduling approach with low computational and communication complexity

The algorithm presented in Section IV-A requires multi-hop communication to gather information about all required transmissions within the network. This has a significant overhead and because it is done before

$$\begin{cases} \delta_j.start \geq \delta_i.start + \delta_i.duration + T(\delta_i.src, \delta_j.src) & \text{if } (\delta_i.dst = broadcast \text{ or } \delta_j.dst = broadcast) \text{ and } Interfer(\delta_i.src, \delta_j.dst) \\ \delta_j.start \geq \delta_i.start + \delta_i.duration + max(& \\ \quad T(\delta_i.src, \delta_i.dst) - T(\delta_j.src, \delta_i.dst), & \text{if } Interfer(\delta_i.src, \delta_j.dst) \\ \quad T(\delta_i.src, \delta_j.dst) - T(\delta_j.src, \delta_j.dst)) & \\ \delta_j.start \geq \delta_i.start & \text{otherwise} \end{cases} \tag{5}$$

Fig. 5. Extended set of simplified scheduling constraints allowing broadcast scheduling.

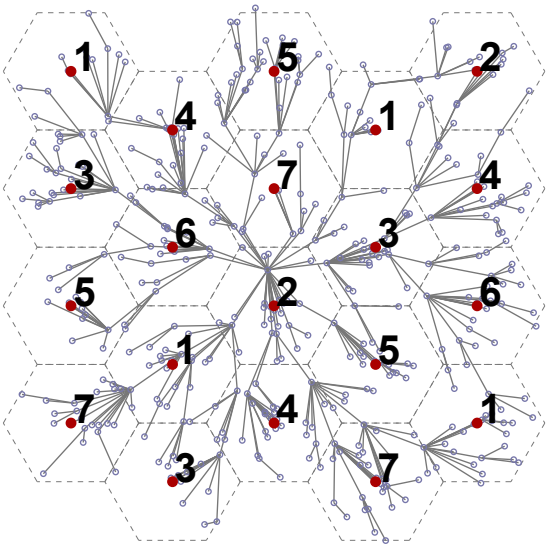


Fig. 7. Example of a deployment.

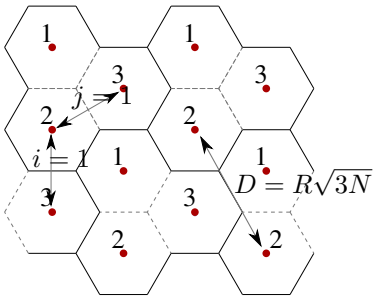


Fig. 8. Cellular network example.

scheduling, this communication will be done in an unscheduled way.

To reduce this communication overhead, we propose a distributed scheduling approach based on a clustering concept. We propose a technique in which cluster-heads are time-schedule arbiters for a cluster and nodes will send a request to the cluster-head to do a communication. The clusters are assigned a timeslot, which can span up to several seconds and will schedule all the requested transmissions in their timeslot. The timeslots can be reused in other clusters and this will ensure that minimal interference occurs between clusters.

Figure 7 illustrates an example deployment setup. The cluster-heads are in the center of their cluster and the numbers shown in the cluster indicate the used timeslot of the cluster. The small dots are sensor nodes scattered across the complete deployment area and the lines between nodes indicate communication links. Communication does not necessarily have to be done from or towards the cluster-heads and can be done to any node within the communication range. The links are set up in such a way that information is collected at a central sink.

The size of the clusters is dependant on the communication range of the nodes. We assume that all nodes in the network use the same output power and carrier frequency for transmissions and will therefore have the same communication range. All nodes within the cluster should be able to communicate with the cluster-head, therefore the cluster size should not be bigger than the communication range. We assume the radius of the cluster is exactly the size of the maximum communication range. The actual size can be calculated using the path loss expressed in Equation (3).

The clusters in our approach are similar to cells in a cellular network. If we assume that the shape of a cluster in our approach is hexagonal, we can then use the equations from cellular networks to calculate the number of timeslots required. The number of timeslots determines the reuse distance. One may recall that the reuse distance is the minimum distance between two clusters that share the same timeslot, see Figure 8 for an example where the number of timeslots can not arbitrarly be chosen and is determined from the following formula:

$$N = i^2 + ij + j^2 \tag{6}$$

The i and j parameters determine the reuse distance of a timeslot along two axises. The reuse distance (D) can be calculated from the number of cells per cluster (N) and the cell radius (R):

$$D = R\sqrt{3N} \tag{7}$$

The reuse distance is the minimum distance between two interfering senders in the network. The larger the distance between two interferers, the lesser interference will be experienced during communication. If a total of 3 timeslots are used, the closest distance between two interfering nodes is exactly the radius of the cluster. If more timeslots are used, the distance between two

	Cluster						
	1	2	3	4	5	6	Max
Slot 1	1.33			1.57			1.57
Slot 2		1.61			1.43		1.61
Slot 3			1.37			1.45	1.45
Slot length	1.57	1.61	1.45	1.57	1.61	1.45	

Fig. 9. Results of calculating slot length based on cluster schedule lengths.

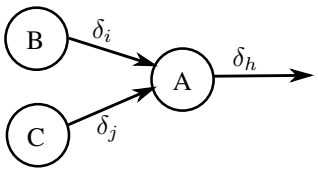


Fig. 10. Multihop scheduled communication.

interfering nodes will be larger, resulting in less noise from neighbouring clusters.

The nodes within a cluster all register their transmissions to the closest cluster-head. The cluster-head is therefore able to schedule all the transmissions within its cluster. After doing so, it will send the minimum length of its local schedule to the central cluster-head. The central cluster-head will assign timeslots to the clusters and determine the length of each timeslot. The timeslots do not necessarily have to be of equal time. The central cluster-head will assign the maximum schedule length of all clusters that share the same timeslot.

The cluster-heads will determine the order of transmissions within their cluster. This can be done using different optimization criteria as presented in [7]. We will be using the greedy approach in which transmissions are scheduled based on the minimum delay.

For scheduling the transmissions within a cluster we can use the algorithm from [7] or the reduced complexity algorithm from Section IV-A. The algorithm presented in Section IV-B will yield a smaller computational and memory space complexity, but because the number of transmissions per cluster is in practice limited, the algorithm presented in Section IV-A may as well be a good option.

Figure 9 shows an example of how the algorithm works. The table shows for all clusters the calculated cluster schedule lengths. The cluster-head schedules all transmissions within its cluster and determines the clusters schedule length. The central cluster-head determines the maximum of all schedule lengths per slot and assigns the maximum schedule length to the slot. The schedule length and slot lengths are then the only information the central cluster-head needs to communicate to the other cluster-heads.

C. A centralized scheduling approach with transmission dependencies

The following algorithm is an extension of the approach shown in Section IV-A. This approach uses dependencies to force an ordering in the scheduling of transmissions. This can be used to reduce the average end-to-end delay for sending packets over multihop connections. An example in which dependencies can be used is shown in Figure 10. In this scenario two nodes (A and B) are sending their packets towards node C and node C is forwarding the (aggregated) packet. In this example it would be benificial for the end-to-end delay to first schedule the transmissions from A and B in such a way that they are received by C before C starts transmitting. If C receives the packets from A and B after it has transmitted its packet it will have to wait for a new iteration of the schedule to forward the packets from A and B.

To add scheduling to the algorithm of Section IV-A, we first have to define a list of dependencies. We store the dependencies in a list D and every dependency is a tuple defining a dependency between two transmissions. In the example shown above, transmission δ_h will have to be transmitted after the reception of transmission δ_i and δ_j. The dependency list D therefore constains tuples (δ_i, δ_h) and (δ_j, δ_h).

Our algorithm for scheduling with dependencies is shown in Figure 11. When scheduling with dependencies, the options of possible transmissions are limited by the dependencies. When a transmission still has dependencies it is considered blocked and can not be scheduled. This transmission still has a tuple in the dependency list D. Only the transmissions without dependencies are considered as next transmission and from all these transmissions we greedily select the next transmission. Once we schedule a new transmission we can remove the tuples of dependent transmissions from the dependency list D, thereby freeing or unblocking possible new transmissions.

The first transmission we schedule is a transmission which has no dependencies. To find this transmission we go through the list of transmissions and find an entry which has no dependency entry in the list D, we do this by finding a transmission δ, which does not have any tuple in D: $(*, \delta) \notin D$. After this, we schedule the first transmission and remove all dependencies for this transmission. Once we have selected the first transmission, we start scheduling the minimum next transmission with no dependency. While determining the minimum transmission to be scheduled we only consider transmissions which have no dependencies. Once the transmission is scheduled, we remove all dependencies related to this transmission. This frees up new transmissions which can be considered during a next round of the scheduling algorithm. We continue until all transmissions are scheduled. Because at every round we consider only transmissions which have no dependencies in the dependency list D, we ensure an ordering of the transmissions.

The dependencies can be derived from the routing algorithm. For example in a data-collection network all data is routed towards a lower hop node until it reaches a single central node in the network. When the parent for a node is selected by the routing algorithm, a transmission is generated from the node to the parent. Next to the transmission also a dependency for this transmission with

```
V ← transmissions {Set of all transmissions}
D ← dependencies {Set of all dependencies}
schedule ← [N] = 0 {Resulting schedule}
{Find the first transmission that can be scheduled}
for δ ∈ V do
  if (∗, δ) ∉ D then
    index ← δ_index
  end if
end for
schedule[index] = 0 {Schedule the first transmission}
time = 0
last = index
D ← D \ (δ, ∗) {Remove all dependencies}
V ← V \ δindex {Remove transmission from set}
{Scheduling loop schedules transmissions greedy}
while !empty(V) do
  time_min ← infinity
  {Calculate minimum starting time for remaining transmissions}
  for δ ∈ V do
    schedule[δ_index] = max(schedule[δ_index], time + constraint(δ_last, δ_index))]
    {See if this transmission has the smallest starting time}
    if (∗, δ) ∉ D and schedule[δ_index] < time_min then
      time_min ← schedule[δ_index]
      index ← δ_index
    end if
  end for
  {Schedule transmission with smallest starting time first}
  time = time_min
  last = index
  V ← V \ δ_index
  D ← D \ (δ, ∗)
end while
```

Fig. 11. Reduced complexity algorithm for scheduling transmissions with dependencies.

the parents transmission should be generated. This causes the transmission to be scheduled from the highest hop nodes first to the lower hop transmissions. Care should be taken that no dependency-cycles are generated, because this renders the dependencies to be unschedulable.

V. EVALUATION OF COMMUNICATION AND COMPUTATION COMPLEXITY

To evaluate the different centralized and distributed scheduling approaches, we will first discuss briefly their complexity in terms of number of communications required as well as computational complexity of different approaches. The complexity overview of all scheduling approaches can be seen in Figure 12.

- Centralized Scheduling: In this case we assume all transmissions as well as position information are collected in a central location. The communication complexity is $n \cdot hops_{avg}$ (The average number of hops), because all transmission information needs to be sent over a multi-hop link to the central scheduler. For scheduling the links we will use the algorithm described in [7], whose complexity is $O(n^3)$.
- Reduced Complexity Centralized Scheduling: This is the algorithm described in Section IV-A. The computational complexity of this algorithm is $O(\frac{1}{2}n^2)$. The communication complexity is the same as the other centralized scheduling approach, namely $O(n^3)$.
- Distributed Scheduling: In the distributed situation, the transmissions are sent only to the cluster-head ($O(n)$ communications). The cluster-head will calculate a schedule for its own cluster and will forward the length of its schedule over a multi-hop link to the central scheduler. This results in $O(hops_{avg}k)$ number of communications. On average, the number of transmissions per cluster is n/k, which results in a computational complexity of $O((n/k)^3)$ per cluster, but also for the whole network.
- Distributed Reduced Complexity Scheduling: It is similar to the distributed approach, but the scheduling per cluster uses the reduced complexity centralized scheduling algorithm. This reduces the scheduling algorithm complexity to $O(\frac{1}{2}(n/k)^2)$ per cluster. The communication complexity remains $O(hops_{avg}k)$.

The packet size of all approaches is constant and does not grow with respect to the number of nodes in the network. From the evaluation of the complexity of the different approach, we can see that the distributed approaches have a much lower computational and communication overhead compared to the centralized approaches. The scalability of the distributed approaches is therefore much better than the centralized approaches.

VI. EVALUATION OF SCHEDULING EFFICIENCY

To evaluate the scheduling efficiency of the different approaches, we implement them in c++. We evaluate the algorithms for different sizes of deployments. The parameters can be found in Figure 13(a). The network size ranges from 500 up to 8000 nodes scattered randomly over an area. The communications are set up in such a way that all data is collected at a central sink, similarly to the deployment illustrated in Figure 7.

For the different distributed scheduling approaches a reuse distance should be selected. We evaluated the distributed algorithms with both 3 as well as 7 timeslots. 3 timeslots is the minimum number of timeslots required and the reuse distance in this case will be exactly the interference range. Using 7 timeslots increases the reuse distance beyond the interference range, this provides a guard band for when the interference range in reality can not be that accurately estimated.

The evaluation results are shown in Figure 14(a). We see that the centralized approach performs the best, which is expected. This is due to the fact that the centralized approach has all link and deployment information of the network during the scheduling, while the distributed approach splits up the scheduling in sub-problems and uses local information only. The centralized approach places a lower bound on the achievable schedule length.

The reduced complexity centralized algorithm performs only slightly worse, the difference in schedule lengths is

Scheduling approach	Computational	Communication	Packet size
Centralized	$O(n^3)$	$2(n \cdot hops_{avg})$	$O(1)$
Reduced Complexity Centr.	$O(\frac{1}{2}n^2)$	$2(n \cdot hops_{avg})$	$O(1)$
Distributed	$O((n/k)^3)$	$2(n + k \cdot hops_{avg})$	$O(1)$
Distributed Reduced Complexity	$O(\frac{1}{2}(n/k)^2)$	$2(n + k \cdot hops_{avg})$	$O(1)$

n = Number of transmissions
k = Number of clusters

Fig. 12. Complexity of different scheduling approaches compared.

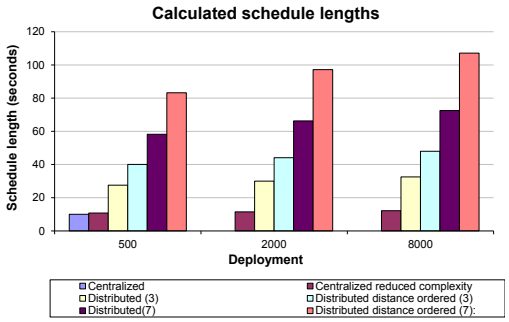
Parameter	Value
Communication range:	500m
Data rate:	1000bps
Propagation speed:	1500 m/s
Node placement:	random / uniform

(a) General parameters

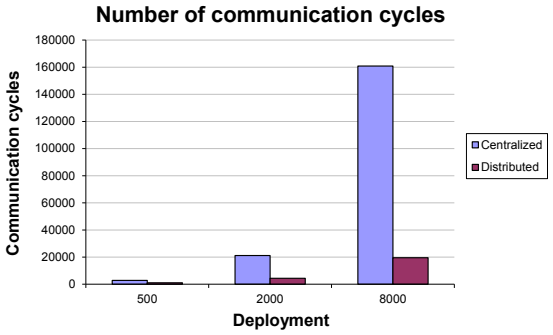
Parameter	Small	Medium	Large
Clusters:	4 x 3	7 x 7	14 x 14
Area size:	3.2 x 3.1km	5.5 x 6.6km	11 x 13km
Nodes:	500	2000	8000

(b) Different deployment sizes

Fig. 13. Simulation parameters.



(a) Schedule length of different scheduling approaches



	Scheduling approach	
Network size	Centralized	Distributed
500	2813	1040
2000	21147	4418
8000	160878	19540

(b) Number of communication cycles required to setup schedule

Fig. 14. Results of simulation for different deployments and scheduling approaches.

only marginal. Therefore the reduced complexity centralized algorithm is a good alternative to the full complexity centralized algorithm. In Section IV-A and Section V we have already shown that the reduced complexity algorithm has large benefits in terms of computation and memory complexity. From the results of the simulation, we can conclude these benefits come at almost no cost in terms of schedule efficiency.

Among the distributed approaches, the distributed approach which minimizes schedule length and uses 3 timeslots, performs about twice as worse as the centralized approach. The approach that orders the transmissions based on distance of the transmission performs worse. The fact that the distributed approach performs worse when the network size increases is because for every timeslot the maximum schedule length from all clusters using that timeslot is used. If more clusters use the same timeslot, the maximum schedule length over all these clusters will go up.

The schedule lengths of the distributed approach are on average 270% of the centralized approach when 3 timeslots are used, and 580% when 7 timeslots are used. This shows that when the scalability, computational and communication benefits are irrelevant a centralized approach is still much preferred.

In Figure 14(b) the amount of communications cycles required to set up the network is shown. The difference between the centralized and distributed approach can be seen quite clearly. The centralized approach does not scale very well to large network sizes and requires large number of communication cycles. The distributed approach grows almost linearly with the size of the network. The number of communication cycles required is a little over 2 times the number of nodes in the network. The packet size of the messages is independant of the number of nodes in the network as has been noted before and contains only position and transmission information, or total schedule length for the cluster heads.

A. End-to-end delay

A criteria for optimization, next to the criteria of optimizing for throughput, may be the time it takes for a packet to travel from the source node to the central node. In this section we will look at this end-to-end delay and we will look at how scheduling dependencies can be used to reduce this end-to-end delay. We have simulated the network in the same setup as before, we have used different number of nodes: 150 nodes, 500 nodes and 2000 nodes. However in this scenario we have setup the transmissions to aggregate the result of the child nodes. Using the shortest hop distance routing algorithm we determine for every node a parent. Every parent will have to send a packet of size 32 bytes plus the number of childs times 32 bytes. So every node is able to send its own data and forward the data from all its childs. The total size of data a parent will have to send depends on

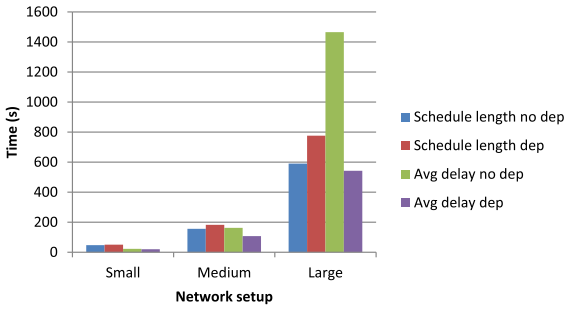


Fig. 15. Scheduling with and without dependencies

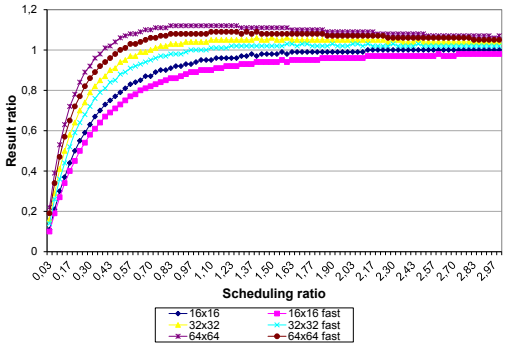


Fig. 16. Scheduling at different ratios

the number of children (n) as follows:

$$size_{total} = 32 + 32 * n \tag{8}$$

In this scenario we look at the length of a single run of the schedule, but we will also look at how long it takes for every packet to travel from the source node to the central node. If no dependencies are used, a packet may be in the network for several subsequent runs of the schedule. If dependencies are used, the parent will wait for all packets to arrive from the children and then starts transmitting his packet and forwards all of his children packets. The results are shown in Figure 15.

What can be seen from these results is that when scheduling with dependencies, the schedule length is increased. This results in a lower throughput. However the average end-to-end delay is decreased because the packets can all be delivered to the central node in a single run. One can see that for the small network the difference in end-to-end delay is not that substantial. This is because in these small networks the number of hops a packet has to travel is small and the length of a schedule run is still short. For medium and large networks the difference is substantial.

Looking at the results, one should consider the application and whether it makes sense to optimize for end-to-end delay. Considering that quite large networks with large number of hops need to be constructed before the difference becomes noticable. One scenario where using dependencies does make sense is when a very low duty cycle is used. The network may sense, send data and then go to sleep for a considerable time. For example the network may sense at a rate of every 10 minutes or every hour. In this scenario it would make sense to optimize for end-to-end delay, because every subsequent run of the schedule may add a delay of 10 minutes or an hour. Packets that require multiple runs of the schedule before being delivered add a delay of many minutes between every iteration of the schedule.

Another scenario for optimizing the end-to-end delay would be when distributed processing such as aggregation is used. In such a scenario a parent can not send before it has received all data from its children. In such a situation the ordering of transmissions is required and the scheduling algorithm with dependencies can be used to achieve a good throughput for the communication in such a network.

B. Scheduling efficiency at different propagation time / modulation time ratio

In this section we evaluate the performance of scheduling independent of the size of the deployment or the modulation rate used at the low-level radio. To do so, we realise that the two most important factors of the performance of the schedule are (I) the propagation time between nodes and (II) the modulation time of the packets. We can simulate the scheduling approach under different sizes of deployments, different packet sizes and different modulation rates, however we can also simulate independent of these parameters by taking the modulation time / propagation time ratio. We define this ratio as follows:

$$ratio_{scheduling} = \frac{time_{modulation}}{time_{propagation}}$$

To give an example, say we want to send 125 byte packets at a modulation rate of 1000bps. The modulation time for a packet in this example is 1 second. Say the average distance between nodes in our network is 1500 meters, this is an equivalent of a propagation time of 1 second. In this scenario our scheduling ratio is 1.

In another scenario in which we want to send 125 byte packets at a modulation of 10kbps, the modulation time for a packet is 100ms. Having an average distance of 150m between nodes results in the same scheduling ratio.

To define a result value independent of the modulation time, we use the following ratio:

$$ratio_{result} = \frac{throughput_{schedule}}{throughput_{radio}}$$

We look at the resulting throughput of the schedule in relation to the throughput available of the radio. This makes the result independent of the chosen radio throughput. We calculate the schedule with different scheduling ratios, for a deployment of nodes uniformly deployed. We simulate this for different number of nodes: 16 nodes, 32 nodes and 64 nodes.

Interesting to see is that there is a certain optimum for the performance around a scheduling ratio of 1. When

going to lower scheduling ratios, the efficiency starts to decrease rapidly. In these scenarios the propagation time takes the overhand in the schedule and the schedule effectively becomes sparse. When going to the higher scheduling ratios than 1, the efficiency of the schedule decreases a little bit but seems to converge to a result ratio of 1.

What can be seen from Figure 16 is that when the propagation time takes the overhand, the efficiency of communication in terms of bandwidth starts to decrease. We have shown this for scheduled MAC protocols but the result may also be valid for unscheduled MAC protocols in the underwater environment. Therefore if the propagation delays are large it makes sense to aggregate more packets into a single transmission or switch to a lower modulation rate. This can be done without losing too much efficiency.

Another observation from Figure 16 is that for different number of nodes the performance slightly differs. We tried to define an indicator for the performance of the scheduling using the modulation time and the propagation time. However from the results it becomes clear that this does not fully describe the performance. The ratio gives a good indication on what the expected performance will be, but when more nodes and therefore more links are in the network the performance does slightly increase. We believe this is because when more links are available the greedily approach of the scheduling algorithm works better because at every step it has more options to choose from.

Using ratios we determined the efficiency of communication scheduling independent of data-throughput of the radio, data packet sizes and node distances. The performance of the schedule is dependent on the number of links. We have shown that communication scheduling achieves the highest throughput when the ratio modulation time and propagation time is 1. This shows that when a certain average distance is dictated by a deployment, the modulation time and data size should be selected accordingly.

C. *Efficiency of broadcast scheduling*

Scheduling broadcasts is possible but is not optimal as scheduling unicasts. To evaluate the efficiency of broadcast scheduling we placed four beacons on a surface and transmit broadcast messages. On the surface there are ten nodes which should receive the broadcast message. We measure the time it takes until all ten nodes have succesfully received all four broadcast messages.

We compare the performance of scheduled communication with ALOHA [6]. We set the sending rate of the beacons to $G = \frac{1}{2}$ which is the optimal sending rate for pure ALOHA [6]. We run the simulation at different distances between beacons and at two different modulation rates. The results are shown in Figure 17.

What can be seen from the results is that scheduling broadcasts is not optimal. This is because when scheduling broadcasts the messages are scheduled very

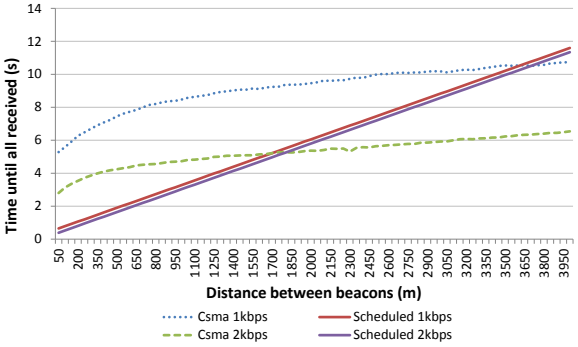


Fig. 17. Result of broadcast scheduling at different beacon distances and different modulation rates

pessimisticly, while in reality collisions occur much less often. The graph shows therefore a cut off point where ALOHA communication performs better than scheduled communication.

From the results it also becomes clear that the performance of scheduled communication is very dependent on the distance between beacons. This is because the delays for transmitting a packet are calculated based on the distance between beacons. The modulation rate, however, does not have a significant impact on the performance. The graph shows two lines for scheduled communication where the higher modulation rate only shows a slight improvement in performance (the 2kbps simulation is the lower line while the 1kbps simulation is the top line).

The ALOHA performance is not very dependent on the distance between the beacons, i.e. the distance is a factor in the performance but the modulation rate is the biggest factor. When the modulation rate doubles the performance of ALOHA almost doubles. The biggest factor in the performance of ALOHA is the modulation rate.

VII. CONCLUSION

Scheduling algorithms for underwater communication allows mitigating the effects of the long propagation delay of the acoustic signal. Scheduling has significant benefits in terms of throughput, energy consumption, and reliability.

In this paper we discussed the extended set of simplified scheduling constraints and introduced a centralized and a distributed scheduling technique for underwater acoustic communication systems. The centralized approach achieves the highest throughput of all scheduling approaches but does this at the cost of high computational and communication overhead.

The distributed approach groups all transmissions together in clusters from which they originate. Nodes within a cluster communicate with the cluster-head only for scheduling their link. Our approach does not place any restrictions on the communication patterns. It does not restrict communication between sink and node and nodes can communicate directly with other nodes within communication range. Each cluster-head will calculate a schedule for its cluster and will forward the total schedule

length of its cluster to a central scheduler. The central scheduler will schedule the timeslots and assign a timeslot to each cluster. Compared to the centralized approach, the distributed approach has a much lower communication and computational overhead.

Comparing communication and computational complexity of the proposed algorithms shows that the distributed approach is much more scalable in larger networks. We also evaluated the schedule lengths of different scheduling approaches. The reduced complexity centralized approach calculates only marginally less efficient schedules, and is therefore a good replacement for the full complexity approach.

We have also introduced an approach to allow scheduling with dependencies between transmissions. This allows to restrict scheduling transmissions in a certain order. We have used this to schedule transmissions from higher hop nodes before transmissions from lower hop nodes. This allows aggregation of data from the outside of the network to a central data collection node and can be used to reduce the end-to-end delay of packets in a data-collection network.

We presented a novel way of estimating the performance of scheduling based on the ratio of modulation time and propagation time. We have shown that this ratio gives a good indication of the expected performance even though the number of links also has a small impact on the performance.

We have also evaluated the performance of broadcast scheduling and compared its performance with ALOHA. We show the performance of broadcast scheduling is dependent on the distance between nodes, while the performance of ALOHA is dependent on the modulation rate. We also show that ALOHA is able to outperform broadcast scheduling. This shows that, as opposed to unicast scheduling, broadcast scheduling is not always a better choice than ALOHA.

Finally we evaluate the end-to-end delay performance of scheduling with dependencies and discuss in what cases dependencies are benificial.

Our future work includes considering the effects of acoustic signal such as refraction, multipath and propagation speed variability on performance. Other effects that will be considered are node dynamics, position estimation errors and time-synchronisation errors. We also want to verify the results of the simulation in real experiments.

ACKNOWLEDGMENT

This work is supported by the SeaSTAR project funded by the Dutch Technology Foundation (STW).

REFERENCES

[1] A. A. Syed, W. Ye, and J. Heidemann, "T-Lohi: A New Class of MAC Protocols for Underwater Acoustic Sensor Networks," in *IEEE INFOCOM*, 2008.

[2] M. Molins, "Slotted FAMA: a MAC protocol for underwater acoustic networks," in *In IEEE OCEANS06, Singapore*, 2006, pp. 16–19.

[3] C.-C. Hsu, K.-F. Lai, C.-F. Chou, and K. C.-J. Lin, "ST-MAC: Spatial-temporal mac scheduling for underwater sensor networks." in *INFOCOM*. IEEE, 2009, pp. 1827–1835.

[4] P. M. Kurtis Kredo II, "Distributed scheduling and routing in underwater wireless networks," *Globecom 2010*, 2010.

[5] L. F. Vieira, J. Kong, U. Lee, and M. Gerla, "Analysis of aloha protocols for underwater acoustic sensor networks," in *Work in Progess poster at the First ACM International Workshop on UnderWater Networks (WUWNet)*. Los Angeles, California, USA: ACM, September 2006.

[6] A. Tanenbaum, *Computer Networks*, 4th ed. Prentice Hall Professional Technical Reference, 2002.

[7] W. van Kleunen, N. Meratnia, and P. J. Havinga, "A set of simplified scheduling constraints for underwater acoustic mac scheduling," *AINA*, 2011.

[8] W. van Kleunen, N. Meratnia, and P. J. Havinga, "Mac scheduling in large-scale underwater acoustic networks," in *8th International Joint Conference on e-Business and Telecommunication, ICETE 2011, Sevilla, Spain*. USA: IEEE Computer Society, July 2011, pp. 27–34.

[9] W. van Kleunen, N. Meratnia, and P. J. Havinga, "Scheduled mac in beacon overlay networks for underwater localization and time-synchronization," in *The Sixth ACM International Workshop on Underwater Networks (WUWNet) , Seattle, USA*. New York: ACM, December 2011, p. 6.

[10] A. A. Syed and J. Heidemann, "Time synchronization for high latency acoustic networks," in *In Proc. IEEE InfoCom*, 2006.

[11] V. Chandrasekhar, W. K. Seah, Y. S. Choo, and H. V. Ee, "Localization in underwater sensor networks: survey and challenges," in *WUWNet '06: Proceedings of the 1st ACM international workshop on Underwater networks*. New York, NY, USA: ACM, 2006, pp. 33–40.

[12] D. E. Lucani, M. Stojanovic, and M. Médard, "On the relationship between transmission power and capacity of an underwater acoustic communication channel," *CoRR*, vol. abs/0801.0426, 2008.