FWB: Funneling Wider Bandwidth Algorithm for High Performance Data Collection in Wireless Sensor Networks

Rodrigo C. Tavares, Marcos Carvalho,
Marcos A. M. Vieira, Luiz F. M. Vieira
Universidade Federal de Minas Gerais (UFMG)
Computer Science Department
Belo Horizonte, MG, Brazil
{rodrigoct,marcoscarvalho,mmvieira,lfvieira}@dcc.ufmg.br

Bhaskar Krishnamachari
University of Southern California (USC)
Ming Hsieh Department of Eletrical Engineering
Los Angeles, CA, US
bkrishna@usc.edu

ABSTRACT

Many applications in Wireless Sensor Networks (WSNs) require collecting massive data in a coordinated approach. To that end, a many-to-one (convergecast) communication pattern is used in tree-based WSNs. However, traffic near the sink node usually becomes the network bottleneck. In this work, we propose an extension to the 802.15.4 standard for enabling wider bandwidth channels. Then, we measure the speed of data collection in a tree-based WSN, with radios operating in these wider bandwidth channels. Finally, we propose and implement Funneling Wider Bandwidth (FWB), an algorithm that minimizes schedule length in networks. We prove that the algorithm is optimal in regard to the number of time slots. In our simulations and experiments, we show that FWB achieves a higher average throughput and a smaller number of time slots. This new approach could be adapted for other relevant emerging standards, such as WirelessHART, ISA 100.11a and IEEE 802.15.4e TSCH.

CCS CONCEPTS

• **Networks → Link-layer protocols**; *Sensor networks*;

KEYWORDS

Wireless sensor networks, IEEE 802.15.4e, Scheduling

ACM Reference Format:
Rodrigo C. Tavares, Marcos Carvalho,, Marcos A. M. Vieira, Luiz F. M. Vieira and Bhaskar Krishnamachari. 2018. FWB: Funneling Wider Bandwidth Algorithm for High Performance Data Collection in Wireless Sensor Networks. In *21st ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM '18), October 28-November 2, 2018, Montreal, QC, Canada.* ACM, New York, NY, USA, Article 39, 8 pages. https://doi.org/10.1145/3242102.3242112

1 INTRODUCTION

Wireless Sensor Networks (WSNs) are composed of many sensor nodes capable of sensing, computing, and communicating [15]. These networks have a wide range of applications, including environmental and industrial monitoring, agriculture, health and surveillance.

One of the most common communication paradigms of sensor networks is collecting massive data in a coordinated (many-to-one) approach. In this paradigm, a convergecast is implemented for data collection from a set of sensors. Data travels through a tree-based routing topology toward a common sink.

Reliable data transport is important in WSNs. For instance, a wireless sensor network in a nuclear power plant might be used to detect radiation levels. It is essential that all nodes relay the entirety of data in a reliable manner, from the sensor nodes to the base station [8, 10]. In addition, there are many WSN applications that require transfers of large amounts of sensed data. Applications of this type include volcano monitoring [21, 22]; surveillance applications, which involve acoustic data and images; and real-time monitoring systems, which allow for the collection of images captured by the sensor node.

Bandwidth in WSNs is limited. The unlicensed 2.4-GHz ISM band is used by a variety of devices, standards and applications. The IEEE 802.15.4 standard defines physical (PHY) layers in the unlicensed 2.4-GHz band. A total of 16 channels are available in this band, numbered 11 to 26, each with a bandwidth of 2 MHz and a channel separation of 5 MHz. Thus, IEEE 802.15.4 platforms, such as MICAz and Telos [12], have radios operating on the maximum theoretical bandwidth of 250 Kbps [19], given the 2 MHz bandwidth constraint.

One of the main issues in wireless communication is interference, which causes packet loss and poses a challenge for fast data collection in a wireless sensor network. Protocols, such as Time Division Multiple Access (TDMA), address such issues by eliminating collisions and retransmissions and providing guarantees on completion times. One of the main strategies for fast data collection in these protocols is to maximize the number of concurrent transmissions, resulting in both a greater reuse of time slots and high collection rates, thus ultimately decreasing schedule lengths.

In [6], heuristics are proposed to minimize schedule length. The authors prove that if interference is eliminated, scheduling becomes optimal. However, they use links of a single bandwidth as a model, i.e., links of a bandwidth that is constant to all nodes.

The main contributions of this work are the following. First, we propose an extension to the 802.15.4 standard in order to enable wider bandwidth channels. Second, we propose Funneling Wider Bandwidth (FWB), an algorithm that minimizes schedule length

---

MSWIM '18, October 28-November 2, 2018, Montreal, QC, Canada

ACM ISBN 978-1-4503-5960-3/18/10…$15.00
https://doi.org/10.1145/3242102.3242112

in networks with radios that operate on these wider bandwidths. Third, we prove that FWB is optimal for the number of time slots. Finally, we present simulation results that confirm the decrease in the total number of time slots.

This paper is organized as follows. In Section 2, we describe the theoretical background. In Section 3, we present the proposed extension. In Section 4, we model the problem. In Section 5, we discuss TDMA scheduling on convergecasts. In Section 6, we detail the protocol. In Section 7, we provide a complexity analysis of the message overhead and prove that the algorithm is optimal. In Section 8, we show results of simulations and experiments performed with the implemented algorithm. In Section 9, we summarize and compare the related work. Finally, in Section 10, we present our final thoughts and conclusions.

2 THEORETICAL BACKGROUND

Shannon [16] showed that the system capacity C of a channel perturbed by additive white Gaussian noise is determined by a function of the average received signal power S, the average noise power N, and the bandwidth W. The capacity relationship is given by the equation $C = Wlog_2(1 + \frac{S}{N})$. From this equation, in order to increase capacity C, the system needs to increase the average received signal power S or bandwidth W. The option of increasing S usually requires an increase in the output transmission power P [5]. However, it is not always possible to increase P, because of its limit on maximum output. Furthermore, an increase in P means an increase in both energy consumption and interference to other nodes.

Another option is to increase the bandwidth, W. When doubling the bandwidth, the throughput also doubles, making it possible to increase throughput and decrease schedule length.

3 PROPOSED EXTENSION

Figure 1 shows our proposed extension to the 802.15.4 standard. The x-axis represents the frequency spectrum. The y-axis shows the bandwidth. Each box indicates a channel. The 2 MHz line is the current IEEE 802.15.4 standard.

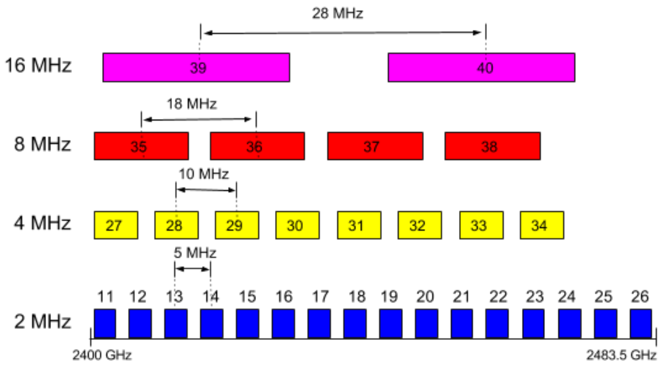


Figure 1: Extension standard with bandwidth channels of 2, 4, 8 and 16 MHz.

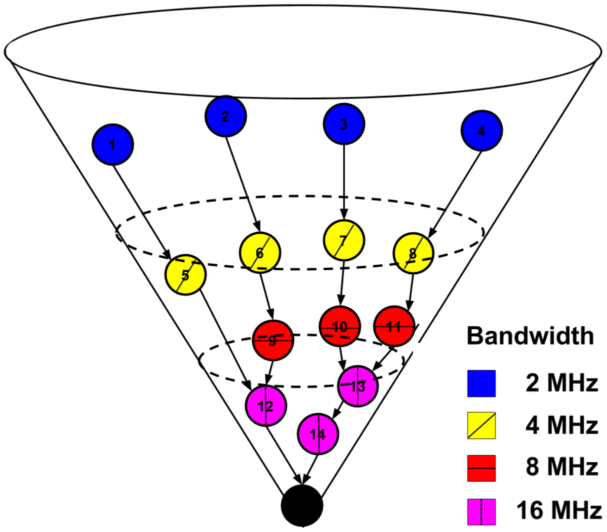


Figure 2: Extension standard with wider bandwidth channels.

With this extension, it is possible to mitigate problems generated by the funneling effect [20], in which traffic near the sink node becomes the network bottleneck, as illustrated in Figure 2. By assigning a wider bandwidth to the nodes near the funnel, throughput can be increased and, as a consequence, the bottleneck is mitigated. In addition, the extension allows these wider bandwidth links to continue functioning normally with the 2 MHz bandwidth links. The bandwidth of the WSN nodes is determined on basis of the number of descendants of the node, which is calculated by Algorithm 1.

Wider bandwidth channels have the disadvantage of being more susceptible to interference, since a wider bandwidth has a larger range of the frequency spectrum. Thus, it is important to allocate the bandwidth in consideration of interference. Moreover, with wider bandwidths, as the number of available "orthogonal" (non-overlapping) channels decreases, the problem of mutual interference becomes more significant. Thus, wider bandwidth channels should be used wisely. Since IEEE 802.15.4 is a technical standard for personal area networks, there might be cases in which bandwidth is locally available.

In the next section, we describe the protocol model and, further, provide an algorithm (Algorithm 1) for allocating the bandwidth of each node.

4 MODELING

We model the WSN as the graph $G = (V, E)$, which consists of a set of sensor nodes, V, that periodically generates data, and a set of edges, E, which represents the wireless links. Links are assumed to be bidirectional. Each edge $e = (u,v)$ consists of nodes u and v. If $e \in E(G)$, then it is said that u is adjacent to v. The vertices of G are denoted by $V(G)$ and the edges by $E(G)$. The node $s \in V$ denotes the sink node. All nodes $u \in V$, with the exception of s, are sources. These nodes generate and transmit packets through a routing tree to sink node *s*. This graph forms a tree on *G*. The spanning tree on *G* rooted at *s* is denoted by $T = (V, E_T)$, where $E_T \in E$ represents the tree edges. A simple model is assumed, in which interference is avoided by using different transmission channels in the links. The network structure is homogeneous. Thus, every sensor node has a half-duplex transceiver that can be configured to operate with a wider channel bandwidth. The network connectivity is static over time, and the wireless link is symmetric. We consider a TDMA protocol where the scheduling frame time is divided into slots. The duration of a time slot allows transmitting exactly one data packet plus a guard interval in order to avoid collisions by synchronization errors. For reasons of simplicity, we use the graph-based protocol model. In this protocol model, the interference range and the transmission range of a node are assumed to be equal. Thus, two links cannot be scheduled simultaneously if the receiver of at least one link is within the range of the transmitter of the other link.

We study raw-data convergecast in the context of periodic data collection, where each node has only one packet to send and the size of each packet is constant. In the same way as [6], we aim to schedule the edges E_T of T by using a minimum number of time slots. However, we also use a variable bandwidth while respecting the following constraint:

Adjacency constraint. Two edges $(i, j) \in E_T$ and $(k, l) \in E_T$ cannot be assigned to the same time slot if they are adjacent to each other. That is, if $\{i, j\} \cap \{k, l\} \neq \phi$, they are adjacent. This constraint is necessary because the transceiver on each node is half duplex. This prevents it from performing simultaneous reception and transmission.

5 TDMA SCHEDULING OF CONVERGECASTS

The problem of minimizing scheduling length for raw-data convergecasts, taking into consideration the interference graph, is proven to be NP-complete by reduction from the known hard problem *Partition Problem* [2]. The optimum lower bounded $\max(2n_k - 1, N)$ is achieved by the algorithm in [17]. However, this optimum is for radios with fixed bandwidth. What would happen if we were to double the bandwidth of a few topology links? The improvement achieved is shown in Figure 3. The schedule length achieved is 4, as the links on the top use twice as much bandwidth, making fewer slots necessary. The packets sent by each slot are shown in the table of Figure 4. Given that we consider data collection to be periodic in raw-data convergecasts, each $e \in E(G)$ is scheduled within each frame only once, and the defined scheduling is repeated in the following frames. Thus, a pipeline is established after a certain frame, and then the sink continues to receive packets from all the source nodes with wider links transmitting more packets per slot. The links between node 1 to sink and node 4 to sink have twice the bandwidth, allowing them to send two packets in a given slot. Starting from frame 2, the sink receives 2 packets in slots 2 and 4, as it has a wider bandwidth. In the same manner, in slot 2, the sink receives 2 packets that were in the buffer of node 1, one at a time. The same occurs for slot 4.

On the other hand, when the bandwidth is three times wider, the number of slots needed is reduced even further, as shown in Figure 5. The packets sent by each slot are shown in the table of Figure 6. Starting from frame 2, the sink receives 3 packets in slots 1 and 2, as it has a wider bandwidth. Thus, in slot 1, it receives 3 packets that were in the buffer of node 1, one at a time. The same occurs for slot 2.

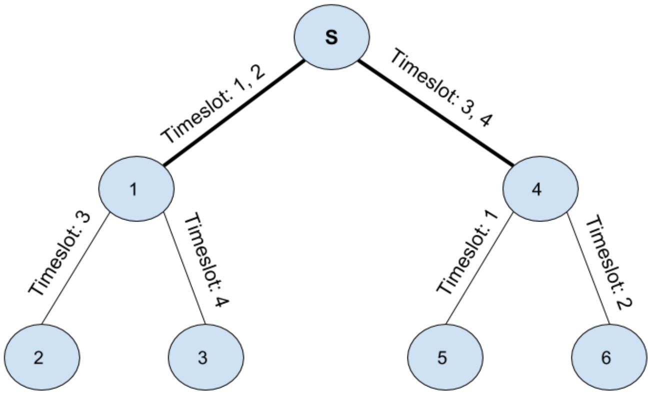
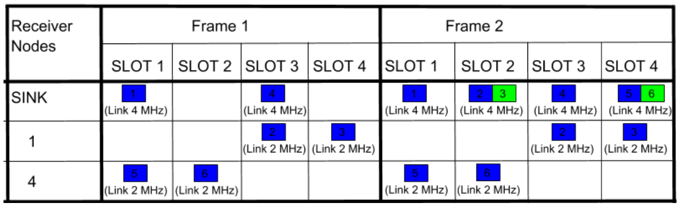


Figure 3: Raw-data convergecast using the proposed algorithm. All of the interfering links are removed in links with dual bandwidth. The numbers on the edges indicate the time slot for each transmission. Schedule length: 4.



Receiver Nodes	Frame 1				Frame 2			
	SLOT 1	SLOT 2	SLOT 3	SLOT 4	SLOT 1	SLOT 2	SLOT 3	SLOT 4
SINK	1 (Link 4 MHz)		4 (Link 4 MHz)		1 (Link 4 MHz)	2 3 (Link 4 MHz)	4 (Link 4 MHz)	5 6 (Link 4 MHz)
1			2 (Link 2 MHz)	3 (Link 2 MHz)			2 (Link 2 MHz)	3 (Link 2 MHz)
4	5 (Link 2 MHz)	6 (Link 2 MHz)			5 (Link 2 MHz)	6 (Link 2 MHz)		

Figure 4: Packets sent in each slot for links with the wider bandwidth of 4 MHz.

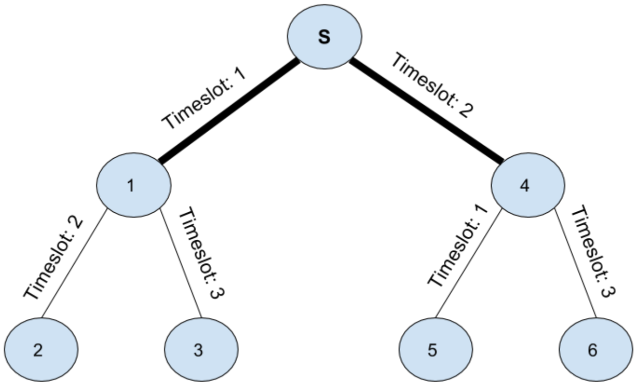
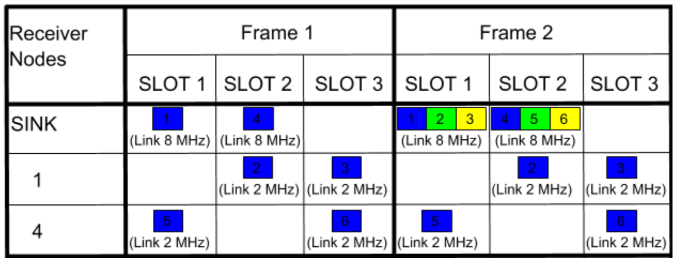


Figure 5: Raw-data convergecast using the proposed algorithm. All of the interfering links are removed in links with triple bandwidth. The numbers on the edges indicate the time slot for each transmission. Schedule length: 3.



Receiver Nodes	Frame 1			Frame 2		
	SLOT 1	SLOT 2	SLOT 3	SLOT 1	SLOT 2	SLOT 3
SINK	1 (Link 8 MHz)	4 (Link 8 MHz)		1 2 3 (Link 8 MHz)	4 5 6 (Link 8 MHz)	
1		2 (Link 2 MHz)	3 (Link 2 MHz)		2 (Link 2 MHz)	3 (Link 2 MHz)
4	5 (Link 2 MHz)		6 (Link 2 MHz)	5 (Link 2 MHz)		6 (Link 2 MHz)

Figure 6: Packets sent in each slot for links with the wider bandwidth of 8 MHz.

In the topology of Figure 5, however, a wider bandwidth would not bring any gain, making it wasteful. The gain that incurs from increasing the bandwidth is limited by the number of descendants of each node.

6 DESIGN OVERVIEW

The purpose of the algorithm is to enable fast data collection by taking advantage of the bandwidths of the variable links. In order to do this, links with many descendants, which constitute the bottleneck of the network, are configured with a wider bandwidth. Thus, non-leaf nodes have a higher allocated bandwidth. The links of the leaves have already been assigned the 2 MHz band. As a consequence, the algorithm calculates the number of descendants of each node before assigning slots in the scheduling. Based on the number of descendants, a wider bandwidth is assigned to the nodes. For instance, in a tree of degree 3, if a node has 3 descendants with only one bandwidth, the node has to allocate 3 time slots to send data from each of their children plus one more for its own data. However, since they are the bottleneck in the network, increasing the bandwidth in three of their links increases channel capacity, which in turn decreases the required number of time slots, as shown in Figure 7. The packets sent by each slot are shown in the table of Figure 8.

The calculation of the number of descendants is done as follows (Algorithm 1). After receiving a request from the sink node, nodes forward the request in broadcast mode. When this message reaches the leaf nodes, they send a reply message to the parents. Then, the parents update a field (count) which determines the number of descendants from this link, and the total number of descendants from the number of reply messages. After that, they send a message to their parent that contains the total number of descendants. Based on the received messages, the parent node calculates the total number of descendants and sends it to its parent. This operation is performed until messages reach the sink node. Now, nodes know the number of descendants they have for each link.

If a node already has x descendants, then it needs $x+1$ bandwidth (for its message) in order to save on the number of slots. Also, if a node has x descendants, then it would be a waste of bandwidth to allocate y bandwidth with $y > x + 1$. Algorithm 2 uses this for optimal slot allocation.

ALGORITHM 1: Calculation of number of descendants

```
⟶ Init
   $count[children] = 0$;
   $descendants = 0$;
   $time = 0$;
   $maxTime = threshold$;
   send message to parent;
⟶ timer expired
   time $*= 2$;
   if (time > maxTime) then
      stop;
⟶ Receive message
   if (count[child] changed) then
      update(count[child]);
      descendants = $\sum$count[child];
      send message to parent;
      reset time;
```

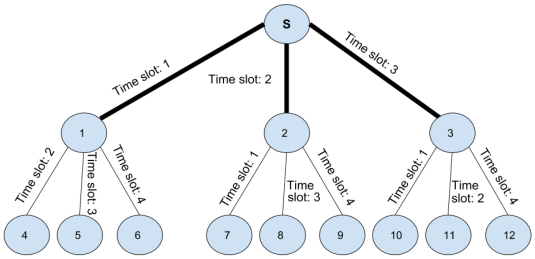
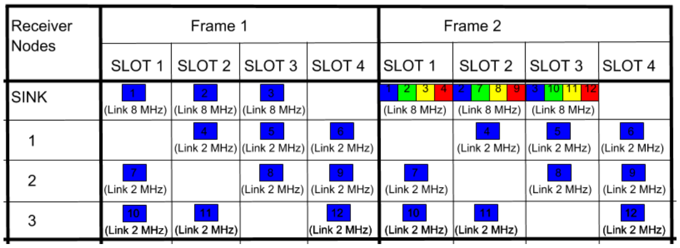


Figure 7: Raw-data convergecast in a tree with 12 source nodes using the proposed algorithm. All the interfering links are removed in triple bandwidth links. The numbers on the edges indicate the time slot for each transmission. Schedule length: 3.



Receiver Nodes	Frame 1				Frame 2			
	SLOT 1	SLOT 2	SLOT 3	SLOT 4	SLOT 1	SLOT 2	SLOT 3	SLOT 4
SINK	1 (Link 8 MHz)	2 (Link 8 MHz)	3 (Link 8 MHz)		1 2 3 4 (Link 8 MHz)	2 7 8 9 (Link 8 MHz)	3 10 11 12 (Link 8 MHz)	
1		4 (Link 2 MHz)	5 (Link 2 MHz)	6 (Link 2 MHz)		4 (Link 2 MHz)	5 (Link 2 MHz)	6 (Link 2 MHz)
2	7 (Link 2 MHz)		8 (Link 2 MHz)	9 (Link 2 MHz)	7 (Link 2 MHz)		8 (Link 2 MHz)	9 (Link 2 MHz)
3	10 (Link 2 MHz)	11 (Link 2 MHz)		12 (Link 2 MHz)	10 (Link 2 MHz)	11 (Link 2 MHz)		12 (Link 2 MHz)

Figure 8: Packets sent in each slot for links with the wider bandwidth of 8 MHz in a tree with 12 source nodes.

Algorithm 2 performs the scheduling. In each iteration, starting from a random node, an edge e is chosen in the Breadth First Search (BFS) order. In order to do this, a chosen node is assigned a bandwidth based on its number of descendants. Then, the node is assigned nTS time slots that are distinct from all its adjacent edges. For instance, in Figure 3, node 4 has 2 descendants, making its workload 3. The available bandwidths for these examples are 2 MHz and 4 MHZ. As the *workload* of node 4 is larger than the *capacity*, it is assigned to the maximum bandwidth. The nTS is calculated based on the number of descendants of the edge e on the tree and on the *factor*. The calculation of the *factor* is based on the gain of the channel capacity. In this instance, *factor* is 2 and nTS is

ALGORITHM 2: FWB

Input: $T = (V, E_t)$

Output: Scheduling and allocated bandwidth to $e \in E_t$

```
1:  while $E_t \neq \phi$
2:    $workload = numberOfDescendants(node) + 1$;
3:    $capacity = maxBandwidth/minBandwidth$;
4:    if ($workload \geq capacity$) then
5:      $allocated\_bandwidth = maxBandwidth$;
6:    else
7:      $allocated\_bandwidth$ = minimum bandwidth available in link
8:      greater than or equal $workload$;
9:
10:   $factor = \frac{allocated\_bandwidth}{original\_bandwidth}$;
11:
12:   $nTS$ = ceil($\frac{workload(node)}{factor}$);
13:
14:    for $i = 1 : nTS$
15:      Is assigned minimum time slot $t$ to edge $e$ respecting adjacency
16:      constraint;
17:      $E_t \leftarrow E_t \backslash e$;
18: end while
```

2. Time slots 1 and 2 are already being used by the adjacent links, so node 4 will transmit to the Source (S) node in time slots 3 and 4.

7 COMPLEXITY ANALYSIS

For complexity analysis, a synchronous communication model will be considered, in which message transmission is done point-to-point. All nodes start to execute the algorithm synchronously, with time divided into slots. Thus, when a message is sent from node u to its neighbor v at time slot t, it must arrive at v before time slot $t + 1$.

We will proceed as follows. First, we enunciate and prove that the algorithm is optimal in regard to the number of time slots. Then, we provide a complexity analysis of the message overhead, by looking into the message and time complexity of Algorithm 1. Finally, we analyze Algorithm 2, which is used for the assignment of wider bandwidth to the nodes near the funnel, thus minimizing schedule length.

THEOREM 7.1. *Algorithm 2 is optimal in regard to the number of time slots.*

PROOF. Suppose that Alg. 2 is not optimal and that there is an algorithm Alg_{OPT} of assignment of time slots with smaller nTS. Given that $nTS = ceiling((workload(node))/factor)$ and the workload is constant, $\mathbf{factor}_{Alg_{OPT}}$ must be greater than $\mathbf{factor}_{Alg.\ 2}$ in order to produce a lower **nTS**. The factor is directly proportional to $bandwidthAllocated$. Therefore, $allocatedbandwidth_{Alg_{OPT}}$ must be greater than $allocatedbandwidth_{Alg.\ 2}$. However, $allocatedbandwidth_{Alg.\ 2}$ is already the largest bandwidth. Therefore, it is not possible for $allocatedbandwidth_{Alg_{OPT}}$ to be greater than $allocatedbandwidth_{Alg.\ 2}$. □

THEOREM 7.2. *Algorithm 1 has a complexity of number of messages of $O(|V|)$ and a time complexity of O(depth(tree)). This complexity is asymptotically optimal in regard to the message and the time.*

PROOF. Note that the algorithm requires a tree topology to have been built by the network before time slot allocation starts, i.e., every node must know its parent in the tree and its number of descendants. Every node begins by sending a message to its parent. After receiving a message from its children, every node must now send a message to its parent, which needs $\Omega(|V|)$ messages. In addition, any message sent by a leaf node in the tree must reach the root, at distance ≤ depth(tree), which needs Ω(depth(tree)) time slots. Thus, message and time complexity is $O(|V|)$ and O(depth(tree)), respectively. □

THEOREM 7.3. *Algorithm 2 requires Ω(depth(tree)) time slots and has a time complexity of $O((|E| + |V|)^2)$.*

PROOF. BFS has a time complexity of $O(|V| + |E|)$. In each iteration of Algorithm 2 (lines 2-13), an edge is chosen e in Breadth First Search (BFS) order, starting from a random node. A bandwidth is assigned to this chosen node based on its number of descendants. Then, after the allocation, there is a broadcast procedure, whereby a message (which includes time slot allocation information) must be sent to every node by the respective parent, which needs $\Omega(|V|)$ messages. Furthermore, the message sent by the root must reach every node at depth(tree) hops away, which needs Ω(depth(tree)) time slots. □

8 RESULTS

8.1 Simulations

We performed simulations with binary trees (balanced and unbalanced) and random trees. For the first experiment, n full complete binary trees were created. For the second experiment, n unbalanced binary trees were created. Finally, in the random topology, n nodes were distributed uniformly at random on a plane field of dimension 1000 *x* 1000.

For each n, we generated 35 instances in order to perform the experiments. To the best of our knowledge, there is no related work that uses wider bandwidth to collect data. Having that in mind, we provide a baseline for comparison. We compared our results with a baseline of 2 MHz, i. e., when a single bandwidth is used in the network, which is the current state-of-the-art.

Figure 9 shows results for the balanced trees. When the links between all nodes, with the exception of the leaves, are twice as wide, the number of slots needed is reduced by half plus one. On the other hand, when the bandwidth available is twice and three times wider, the number of slots needed is reduced by three plus one. Lastly, when the bandwidth is twice, three and four times wider, the number of slots needed is reduced by four plus one. Bandwidth is allocated according to the number of descendants. Figure 10 shows results for unbalanced trees. Results were similar to the previous experiment. A wider bandwidth was found to reduce the number of time slots.

Figure 11 shows the results of experiments performed for instances of randomly generated trees with a confidence interval. The results show that the wider the bandwidth, the smaller the number of slots required. As seen in this result, an increase in the number of nodes does not necessarily imply an increase in the number of time slots. As discussed in [6], the rate of data collection is quite limited by the topology of the routing tree.

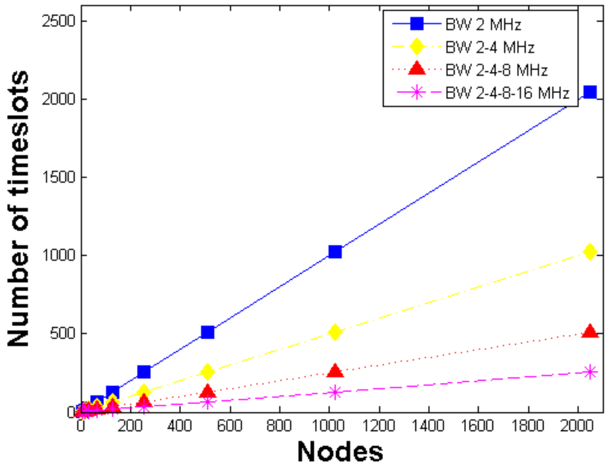


Figure 9: Balanced binary tree

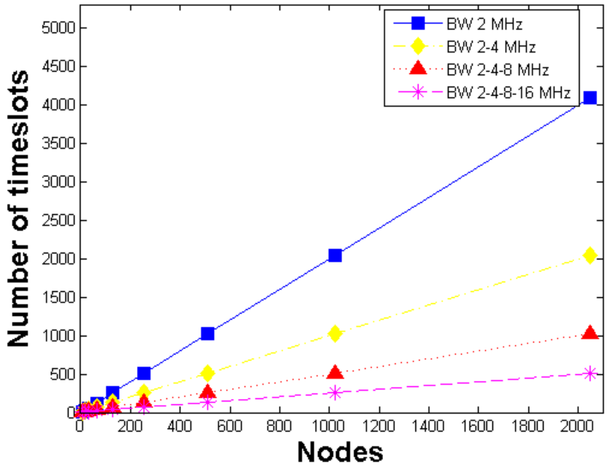


Figure 10: Unbalanced binary tree

8.2 Experiments

We performed our experiments on the FUTEBOL[1] testbed, located at UFMG (Federal University of Minas Gerais), using TinyOS for performance analysis. TinyOS is a lightweight operating system designed for low power wireless sensors. Its design focuses on low power consumption operations [9]. The experiments were performed on the topology shown in Figure 12. We performed 33 experiments for each setting.

TelosB nodes only have a fixed bandwidth. Thus, in order to show the gain provided by the FWB algorithm in this topology, we emulated wider bandwidths: node 0 was emulated to have 4 radios, each operating at a different frequency, so as to make synchronous transmissions possible. This is comparable to assigning a bandwidth 4 times greater to node 0. Thus, we arranged 4 nodes together in order to emulate node 0, while arranging 2 nodes together in order to emulate nodes 1 and 2.

The number of necessary slots is therefore reduced to the topology shown in 13. The schedule length achieved is 2, since nodes 3 and 4 can send the same time slot to node 1, as they are able to send to different radios.

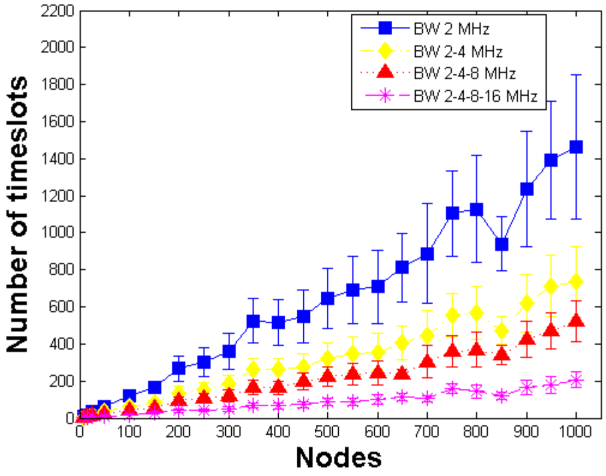


Figure 11: Random tree

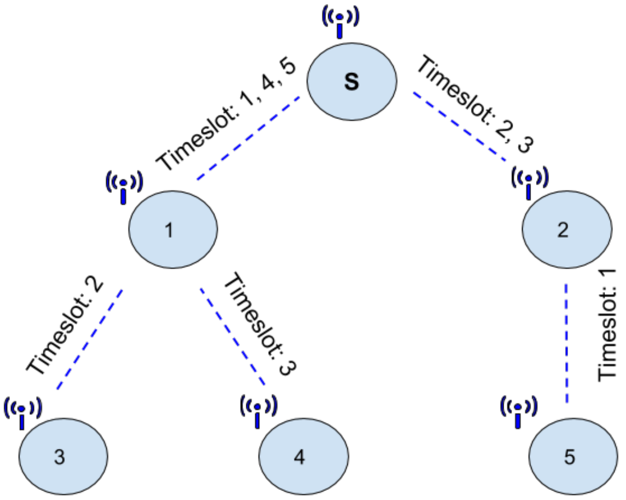


Figure 12: Topology used in experiments with TelosB nodes. Schedule length: 5.

Figure 14 compares the number of packets received per second through the sink node in the topologies of Figure 12, which has a single bandwidth, and Figure 13, which emulates links with wider bandwidth. In the latter, throughput to the nodes is greater, as can be seen in the figure. This is due to a smaller schedule length, which results in more packets being transmitted in a given time interval. However, throughput is not four times higher, as expected. This is

---

[1] http://futebol.dcc.ufmg.br/

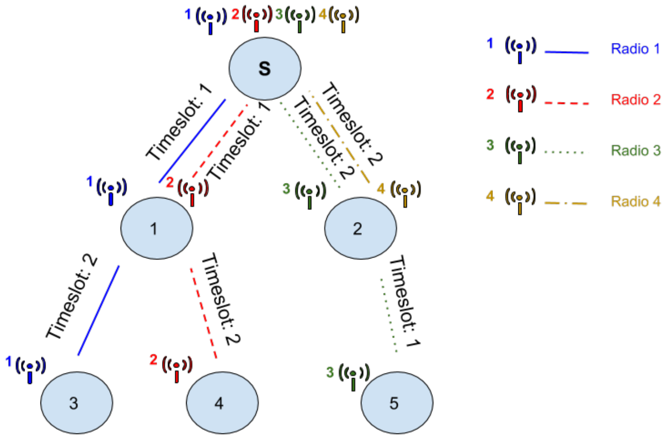


Figure 13: Topology used in experiments emulating wider bandwidth with TelosB nodes. Schedule length: 2.

due to the guard interval setting, which was not sufficiently tuned to prevent simultaneous transmissions and receptions, causing a great deal of losses and, consequently, a lower throughput.

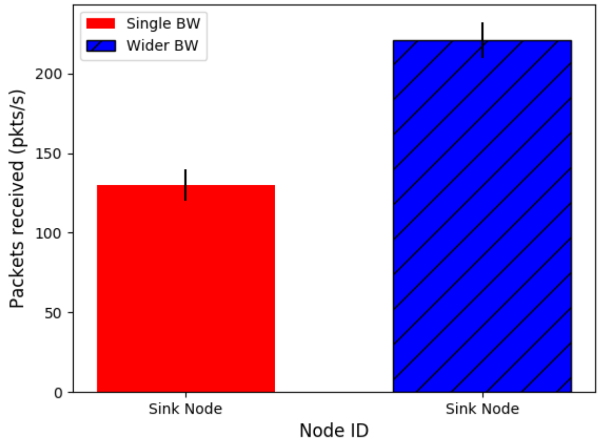


Figure 14: Number of packets per second received by sink nodes in the topologies of Figure 12 and Figure 13.

9 RELATED WORK

Many protocols have been proposed in the literature to improve throughput. CodeDrip [14] uses Network Coding to improve data dissemination but does not propose channel allocation. FlushMF [18] uses multiple frequencies to improve network throughput without taking wider bandwidth into consideration. PIP [13] and P^3 [3] are transport protocols that use multiple channels. These protocols use a TDMA schema for synchronizing transmissions over the pipeline. In P^3, the authors discuss the use of multiple frequencies and multiple paths to increase end-to-end throughput. The combination of these two techniques allows the maximum flow rate in an ideal model (without interference) to be 100% of the channel capacity. To this end, the package pipeline and multiple paths are used. The packet pipeline technique in end-to-end transmissions allows the maximum amount of intermediate transmitters between the source and the destination node for the transmission of packets simultaneously, so as to not interfere with one another and cause packet loss. Through the use of multiple paths, the source node can transmit packets at all times, each by a given channel in a given path. Conversely, the destination node can receive packets at all times, thus allowing the maximum theoretical flow to be 100% with the full pipeline.

In [1], a hybrid CSMA/TDMA MAC protocol is implemented. In order to mitigate the funneling effect, pure CSMA/CA operates network-wide, whereas TDMA scheduling is used in the intensity region of the event funnel, with the result of improving throughput.

Therefore, minimizing schedule length is required to increase the throughput. Many protocols focused on TDMA schedules have been proposed, such as [4] and [6]. In [4], the authors propose a distributed convergecast scheduling algorithm (DCSA) that requires at most $3N$ time slots, where N represents the number of nodes in the network. They prove that the lower bound on the number of time slots required to complete convergecast in a line topology is $3N - 3$. They further prove that if N represents the number of nodes in the network and n_k represents the maximum number of nodes in a branch, then the lower bound on the number of time slots required for convergecast scheduling in multi-line networks is given by $max(3n_k - 3, N)$. Moreover, the number of time slots required by tree topologies is given by $max(3n_k - 1, N)$.

In [7], the authors propose a channel assignment method called RBCA, where they statically assign the channels to the receivers. In this manner, they are able to remove many interfering links, which results in a smaller schedule length.

In [6], a convergecast scheduling algorithm named JFTSS is proposed, where channel scheduling is coupled with time slot scheduling. JFTSS offers a greedy joint solution for constructing a maximal schedule, so that a schedule is said to be *maximal* if it meets the adjacency and interfering constraints, and no more links can be scheduled for concurrent transmissions in any time slot and channel without violating these constraints. Moreover, the authors prove that if all the interfering links are eliminated, the schedule length for one-shot raw-data convergecast is lower bounded by max(2n_k - 1 , N). One-shot data collection refers to the collection of data generated by some asynchronous event, in which data is transmitted individually to the sink. Further, the authors propose an algorithm (LOCAL-TIMESLOTASSIGNMENT) for the assignment of these time slots and prove that when the interfering links are eliminated, the schedule length achieved by this algorithm is max(2n_k - 1 , N).

In [23], the authors study the convergecast scheduling tree with multiple channels (TCMC) problem. They derive an integer programming-based optimal solution to the min length and buffer size scheduling, as well as the min length and channel number scheduling. Moreover, they propose and implement 4 heuristics (4H) for optimal configuration of the number of channels and the topology of the routing tree of a WSN.

In [11], a minimal delay scheduling (MDS) problem is described for finding a tree in a graph and an interference-free slot and channel assignment, so that the convergecast latency is minimized. They prove that the problem is NP-complete and propose a heuristic algorithm (HA) for solving it, which contains three phases. The tree formation phase connects nodes by the shortest path tree with constrained degrees. The slot assignment phase assigns slots to links to achieve optimal latency (regardless of interferences). The channel assignment phase assigns frequency channels to nodes in order to eliminate interferences between links. Simulation and implementation results show that the protocol reduces the convergecast latency in WSNs with multiple channels.

Table 1 is a summary of the scheduling protocols proposed in the literature.

Table 1: Summary of related protocols

Protocol	Centralized	Topology	Wider Bandwidth
DCSA [4]	✗	Line	✗
BFS-TIMESLOT ASSIGNMENT [6]	✓	Tree	✗
LOCAL-TIMESLOT ASSIGNMENT [6]	✓	Tree	✗
RBCA [6]	✓	Tree	✗
JFTSS [6]	✓	Tree	✗
4H [11]	✓	Tree	✗
HA [23]	✓	Tree	✗
FWB	✓	**Tree**	✓

10 CONCLUSIONS

In this paper, we have presented an optimal algorithm for the number of time slots to minimize the schedule length by considering a 802.15.4 extension, which enables channels with wider bandwidth. The main idea is that enabling a wider bandwidth makes the transmission of more data in a given slot possible. Thus, schedule length is minimized and network throughput is increased. Our results have shown a decrease in the number of necessary time slots for collecting data from each node. Our contribution for a wider bandwidth could be adapted and used with other relevant wireless emerging standards, such as IEEE 802.15.4e TSCH and WirelessHART.

For future work, we plan to implement the protocol with the use of an Universal Software Radio Peripheral (USRP).

ACKNOWLEDGMENTS

The authors would like to thank the research agencies CNPq, CAPES, and FAPEMIG for their financial support.

REFERENCES

[1] Gahng-Seop Ahn, Se Gi Hong, Emiliano Miluzzo, Andrew T. Campbell, and Francesca Cuomo. 2006. Funneling-MAC: A Localized, Sink-oriented MAC for Boosting Fidelity in Sensor Networks. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems (SenSys '06)*. ACM, New York, NY, USA, 293–306.

[2] Hongsik Choi, Ju Wang, and Esther A. Hughes. 2009. Scheduling for information gathering on sensor network. *Wireless Networks* 15, 1 (01 Jan 2009), 127–140. https://doi.org/10.1007/s11276-007-0050-9

[3] Manjunath Doddavenkatappa and Mun Choon Chan. 2014. P3: A Practical Packet Pipeline using synchronous transmissions for wireless sensor networks. In *Information Processing in Sensor Networks, IPSN-14 Proceedings of the 13th International Symposium on*. IEEE, 203–214.

[4] Shashidhar Gandham, Ying Zhang, and Qingfeng Huang. 2006. Distributed minimal time convergecast scheduling in wireless sensor networks. In *Distributed Computing Systems, 2006. ICDCS 2006. 26th IEEE International Conference on*. IEEE, 50–50.

[5] Olga Goussevskaia, Luiz F.M. Vieira, and Marcos A.M. Vieira. 2016. Wireless scheduling with multiple data rates: From physical interference to disk graphs. *Computer Networks* 106, Supplement C (2016), 64 – 76. https://doi.org/10.1016/j.comnet.2016.06.016

[6] Ozlem Durmaz Incel, Amitabha Ghosh, Bhaskar Krishnamachari, and Krishna Chintalapudi. 2012. Fast data collection in tree-based wireless sensor networks. *IEEE Transactions on Mobile computing* 11, 1 (2012), 86–99.

[7] Ozlem Durmaz Incel, Amitabha Ghosh, Bhaskar Krishnamachari, and Krishna Kant Chintalapudi. 2008. Multichannel scheduling for fast convergecast in wireless sensor networks. (2008).

[8] Sukun Kim, Shamim Pakzad, David Culler, James Demmel, Gregory Fenves, Steven Glaser, and Martin Turon. 2007. Health monitoring of civil infrastructures using wireless sensor networks. In *Information processing in sensor networks, 2007. IPSN 2007. 6th international symposium on*. IEEE, 254–263.

[9] Philip Levis and David Gay. 2009. TinyOS Programming.

[10] Ruizhong Lin, Zhi Wang, and Youxian Sun. 2004. Wireless sensor networks solutions for real time monitoring of nuclear power plant. In *Intelligent Control and Automation, 2004. WCICA 2004. Fifth World Congress on*, Vol. 4. IEEE, 3663–3667.

[11] Meng-Shiuan Pan and Yi-Hsun Lee. 2016. Fast convergecast for low-duty-cycled multi-channel wireless sensor networks. *Ad Hoc Networks* 40 (2016), 1–14.

[12] Joseph Polastre, Robert Szewczyk, and David Culler. 2005. Telos: Enabling Ultra-low Power Wireless Research. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks (IPSN '05)*. IEEE Press, Piscataway, NJ, USA, Article 48.

[13] Bhaskaran Raman, Kameswari Chebrolu, Sagar Bijwe, and Vijay Gabale. 2010. PIP: A connection-oriented, multi-hop, multi-channel TDMA-based MAC for high throughput bulk transfer. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*. ACM, 15–28.

[14] Nildo dos Santos Ribeiro Júnior, Rodrigo C. Tavares, Marcos A.M. Vieira, Luiz F.M. Vieira, and Omprakash Gnawali. 2017. CodeDrip. *Ad Hoc Netw.* 54, C (Jan. 2017), 42–52. https://doi.org/10.1016/j.adhoc.2016.09.023

[15] Linnyer Beatrys Ruiz, Luiz Henrique A Correia, Luiz Filipe M Vieira, Daniel F Macedo, Eduardo F Nakamura, Carlos MS Figueiredo, Marcos Augusto M Vieira, Eduardo Habib Bechelane Maia, Daniel Câmara, Antonio AF Loureiro, et al. 2004. Architectures for wireless sensor networks. *Proceedings of the 22nd Brazilian Symposium on Computer Networks (SBRC 04)* (May 2004), 167–218.

[16] C. E. Shannon. 1948. A Mathematical Theory of Communication. *Bell System Technical Journal* 27, 3 (1948), 379–423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

[17] Wen-Zhan Song, Fenghua Yuan, Richard LaHusen, and Behrooz Shirazi. 2007. Time-optimum packet scheduling for many-to-one routing in wireless sensor networks. *The International Journal of Parallel, Emergent and Distributed Systems* 22, 5 (2007), 355–370.

[18] Rodrigo C Tavares, Marcos AM Vieira, and Luiz FM Vieira. 2016. Flushmf: A transport protocol using multiple frequencies for wireless sensor network. In *Mobile Ad Hoc and Sensor Systems (MASS), 2016 IEEE 13th International Conference on*. IEEE, 192–200.

[19] M. A. M. Vieira, C. N. Coelho, D. C. da Silva, and J. M. da Mata. 2003. Survey on wireless sensor network devices. In *EFTA 2003. 2003 IEEE Conference on Emerging Technologies and Factory Automation. Proceedings*, Vol. 1. 537–544 vol.1. https://doi.org/10.1109/ETFA.2003.1247753

[20] Chieh-Yih Wan, Shane B. Eisenman, Andrew T. Campbell, and Jon Crowcroft. 2005. Siphon: Overload Traffic Management Using Multi-radio Virtual Sinks in Sensor Networks. In *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems (SenSys '05)*. ACM, New York, NY, USA, 116–129. https://doi.org/10.1145/1098918.1098931

[21] Geoff Werner-Allen, Konrad Lorincz, Jeff Johnson, Jonathan Lees, and Matt Welsh. 2006. Fidelity and yield in a volcano monitoring sensor network. In *Proceedings of the 7th symposium on Operating systems design and implementation*. USENIX Association, 381–396.

[22] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh. 2006. Deploying a wireless sensor network on an active volcano. *IEEE Internet Computing* 10, 2 (March 2006), 18–25. https://doi.org/10.1109/MIC.2006.26

[23] Dingwen Yuan and Matthias Hollick. 2012. Tree-based multi-channel convergecast in wireless sensor networks. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012 IEEE International Symposium on a*. IEEE, 1–9.