2009

Energy-efficient scheduling in WMSNs

Mohamed K. Watfa
University of Wollongong in Dubai, mwatfa@uow.edu.au

Farah Abou Shahla
American University of Beirut

Publication Details

Watfa, M. & Abou Shahla, F. 2009, Energy-efficient scheduling in WMSNs, InfoComp, vol. 8, no. 1, pp. 45-54.

Energy-Efficient Scheduling in WMSNs

MOHAMED K. WATFA[1]
FARAH ABOU SHAHLA[2]

[1]University of Wollongong
Computer Science Department
mwatfa@uow.edu
[2]American University of Beirut
Computer Science Department
fna11@aub.edu.lb

Abstract. Recent technological breakthroughs in ultra-high integration and low-power electronics have enabled the development of tiny battery-operated sensors. The signal processing and communication activities are the main consumers of sensor's energy. Since sensors are battery-operated, keeping the sensor active all the time will limit the battery's lifetime. Therefore, optimal organization and management of the sensor network is crucial in order to perform the desired function with an acceptable level of quality and to maintain sufficient sensor energy for the required mission. Wireless multimedia sensor networks (WMSN) are a new and emerging type of sensor networks producing multimedia content. These networks have the potential to enable a large class of applications. Many such applications require mechanisms to efficiently deliver application level quality of service (QoS) and to map these requirements into network layer metrics such as latency and jitter. Thus an efficient scheduling algorithm that differentiates between levels of services by providing QoS guarantees is needed. Since WMSNs are battery-constrained, the algorithm should take into consideration energy-efficiency as well. In this paper, we present a scheduling algorithm that takes into account the promptness of real-time multimedia streaming. The algorithm divides the frame into slots and assigns the slots to different nodes. We also present an algorithm to interleave the slots in a way to minimize jitter. We have evaluated the interleaving algorithm using Matlab and found that it minimizes the jitter for all nodes in a consistent way while keeping the overhead generated to a certain limit.

Keywords: Sensor Networks, Energy-efficient, QoS, scheduling, WiMax.

(Received October 03, 2008 / Accepted March 24, 2009)

1 Introduction and Motivation

Wireless Multimedia Sensor Networks (WMSNs) are emerging technologies with a wide range of applicability. Their applications include multimedia surveillance that could detect potentially relevant activities like thefts or violations, advanced health care deli-very, automated assistance for the elderly [1], person identification and locating through thermal imaging or face recognition techniques, and many others. Such networks are usually formed of a large number of nodes remaining inactive for a long period of time but becoming unexpectedly active when some action is sensed or detected. Each node is basically characterized by: one or more sensors, embedded processors, low-power radio antennas, and batteries. Different architectures of multimedia networks exist: single-tier flat architecture, single-tier clustered architecture and multi-tier architecture. The single-tier flat architecture contains homogeneous sensors with distributed processing while the clustered architecture contains heterogeneous sensors with centralized processing. Both architectures operate under centralized storage where each cluster or group of sensors is provided with a storage hub. The multi-tier architecture contains heterogeneous sensors with distributed processing and distributed storage. In this approach, some low-power nodes are in charge of simpler tasks while high-power devices with more resources are assigned more complex jobs.

Advantages of WMSNs include the benefit of deploying many nodes in a certain area. These nodes cooperate to perform a certain task and thus allowing for a larger view in case of cameras and perception sensors in addition to multiple viewpoints of the same area. This could be highly beneficial in places where monitoring the area needs constantly moving cameras. Also a set of nodes deployed close to each other will help enhance the view as well as allowing for quicker identification of small changes in the environment (including identification of faces and persons). The inherent characteristics of wireless multimedia sensors allows a node not only to retrieve and store multimedia data but also to process in real-time, correlate and fuse multimedia content received from multiple sources. Thus overlapped sensors can provide different views of the same target while the joint operation (i.e. the processing and fusion of information) of different cameras, audio, infrared sensors or any other heterogeneous nodes can help in clarifying many situations. Another advantage of WMSNs is the advent of multi-resolution views. Fixed cameras lose their resolution properties when zooming on a certain object however with multiple sensors in the area, zooming could be achieved by streaming from two different medium-resolution camera nodes to achieve a high-resolution view of the region of interest.

Many applications of WMSNs require mechanisms to efficiently deliver application level quality of service (QoS) and to map these requirements into network layer metrics such as latency and jitter. Transmission reliability with energy efficiency makes the key for a good design in wireless sensor networks. However, in WMSNs, more light is shed on providing certain QoS guarantees. Most of today's wireless devices are multimedia-oriented and they have as well constraints on energy consumption, size and bandwidth. In multi-hop (or ad-hoc) wireless multimedia sensor network architectures, energy influences node lifetime, and therefore, network lifetime. Thus improving WMSN capabilities include improving energy efficiency in addition to providing good quality of service for multimedia applications.

In this paper, we present a scheduling algorithm that takes into account the promptness of real-time multimedia streaming. The algorithm divides the frame into slots and assigns the slots to different nodes. We also present an algorithm to interleave the slots in a way to minimize jitter. The remainder of this paper will be divided as follows: Section 2 summarizes the major factors influencing the design of multimedia sensor networks; Section 3 presents some of the related work to energy-efficient scheduling in wireless networks. Section 4 presents the proposed scheduling algorithm. In section 5, we explain two interleaving algorithms in order to minimize the jitter. Section 6 evaluates these algorithms using Matlab and provides a comparison between the two and the non-interleaving method. Section 7 provides a conclusion and gives a glimpse of future work.

2 Distinctiveness of WMSNs

The variety of requirements and guarantees offered by wireless multimedia sensor networks, added to their random deployment and architecture, give rise to many factors that influence their design. In the following, some of these factors will be explained in the aim to broaden the understanding of WMSNs.One of the special characteristics of wireless multimedia sensor networks is their high bandwidth demand. Given that the state-of-the-art IEEE 802.15.4 compliant components (such as Crossbow's MICAz and TelosB motes) have a nominal transmission rate of 250 kbit/s, multimedia content requires a transmission bandwidth at least an order higher. Video streaming, in particular real-time streaming, requires higher bandwidth than that supported by currently available sensors with comparable power consumption. Hence, high data rate and low-power consumption transmission techniques need to be managed. In this respect, the ultra wide band (UWB) transmission technique seems particularly promising for WMSNs.

The different applications that are currently being studied for WMSNs, and other applications that are yet to be imagined, entail different requirements. These include simple data delivery modes, still images or snapshots, and audio and video streaming. These different types of applications have different characteristics as well as different QoS requirements. While data delivery modes need no QoS guarantees, snapshots which are event-triggered (for e.g. still images taken each period of time) and multimedia streaming require sustained information delivery. Hence, a strong foundation is needed in terms of supporting high-level algorithms to deliver QoS and consider application-specific requirements. These requirements may pertain to multiple domains and can be expressed, amongst others, in terms of a combination of bounds on energy consumption, delay, reliability, distortion, or network lifetime. As in any traditional wireless sensor network, power consumption is a fundamental concern in WMSNs, and even more than before. In fact, sensors are battery-operated and they are randomly deployed so recharging or changing batteries is a hard task. Add to that, the fact that multimedia applications produce high volumes of data, which require high transmission rates, and extensive processing. While the energy consumption of traditional sensor nodes is known to be dominated by the communication functionalities, this may not necessarily be true in WMSNs. Therefore, protocols, algorithms and architectures to maximize the network lifetime while providing the QoS required by the application are a critical issue.

Furthermore, WMSNs are characterized by their ability to perform multimedia in-network processing on the raw data extracted from the environment. This requires new architectures for collaborative, distributed, and resource-constrained processing that allow for filtering and extraction of semantically relevant information at the edge of the sensor network. One of the advantages of in-network processing is that it may increase the system scalability by reducing the transmission of redundant information, merging data originated from multiple views, on different media, and with multiple resolutions. For example, in video security applications, information from uninteresting scenes can be compressed to a simple scalar value or not be transmitted altogether, while in environmental applications, distributed filtering techniques can create a time-elapsed image. Hence, it is necessary to develop application-independent architectures to flexibly perform in-network processing of the multimedia content gathered from the environment. IrisNet, for instance, uses application-specific filtering of sensor feeds at the source, i.e., each application processes its desired sensor feeds on the CPU of the sensor nodes where data are gathered. This dramatically reduces the bandwidth consumed, since instead of transferring raw data, IrisNet sends only a potentially small amount of processed data. However, the cost of multimedia processing algorithms may be prohibitive for low-end multimedia sensors. Hence, it is necessary to develop scalable and energy-efficient distributed filtering architectures to enable processing of redundant data as close as possible to the periphery of the network.

Other factors that influence the design of wireless multimedia sensor networks include but are not restricted to: multimedia source coding techniques where intra-frame compression and inter-frame compression methods can be used, multimedia coverage where the directivity or direction of acquisition of video sensors differs from their radii of sensing. More on these issues can be found in [1]. Therefore designing a scheduling algorithm for WMSNs should probably take into account all or some of these factors at least. Not to forget that the aim is to achieve energy efficient communication, i.e., allocate transmit powers to different transmitters in such a way that QoS constraints like throughput, delay, jitter, error rate are met by expending minimum possible power.

While a lot of research has been done on some important aspects of WSNs such as architecture and protocol design, energy conservation, and locationing, supporting Quality of Service (QoS) in WSNs is still a largely unexplored research field. This is mainly because WSNs are very different from traditional networks. Thus far, it is not entirely clear how to properly describe the services of WSNs, much less to develop approaches for QoS support.

3 Related Work

A primary concern while scheduling real-time multimedia applications in wireless networks is energy efficiency, since a majority of wireless devices operate on batteries that need to be regularly recharged from a power source. The MAC layer responsible of scheduling messages of different types differs in WMSNs from the traditional networking model. This is because not only QoS requirements are the concern, but also power issues should be taken into account. Many researches in this area have been done however none of the research found is directly addressed towards wireless multimedia sensor networks. In this section, a few of the proposed architectures and MAC protocols of the literature will be reviewed.

Alghamdi, Xie and Qin [3] introduced PARM, a Power-Aware Real-time Message scheduling algorithm developed in aims of meeting time constraints while minimizing the power consumption for real-time wireless networks. The architecture of the PARM strategy is composed of two main modules and a scheduler. The first module is the Admission Controller: its main functionality is to control the flow and reject those packets that do not have a feasible deadline (i.e. their earliest transmission time added to their minimal transmission time is greater than their expected deadline). The accepted packets are scheduled in the Accepted Queue using Earliest Deadline First scheduling algorithm. The packets are then passed to the Energy Consumption Controller module that will attempt to minimize the energy using an algorithm that loops over all arriving messages. For each arriving message, it computes the earliest transmission start time which is the time to continue transmitting the message in hand plus the time to transmit all the messages with earlier deadline. If the deadline of the arriving message is feasible (i.e. it is greater than the sum of the earliest transmission start time and the minimum transmission time) the message will be accepted otherwise it will be dropped by setting the transmission rate of the message to 0. The accepted message will be given the minimal transmission rate by looping on all possible scheduling of the message.The PARM algorithm has demonstrated significant performance improvements however, for multimedia, large-scale networks are typically envisaged and thus the work should have taken into account the scalability factor of WMSNs. With large-scale networks and real-time messaging of multimedia content, it is inefficient to choose the schedule of a packet with the minimal consumption out of the set of all possible schedule decisions. Furthermore, in the paper, the authors assume that the message's feasible deadline should be greater than its earliest transmission time added to its minimal transmission time. Thus a message can be dropped merely because there are too many messages ahead of it with an earlier deadline but not as important as the dropped message is. Here, the notion of priority should be inspected. So why drop a higher priority packet or message just because there are too many scheduled messages with a hotter deadline but with a lower priority.

The higher priority message could actually be scheduled (in most cases that is) before the other messages and still make room for the less priority messages to be transmitted before their deadlines. Another thing is: during a multimedia communication session between two nodes and the main node (BS), each node might be sending different types of multimedia content. The first could be sending live camera broadcast while the second could be updating the database with new still/static images (snapshots). The need for higher bandwidth and less jitter is obviously for the first node. The transmission of the second node's message, although having a critical deadline, should not jeopardize the first node's transmission, hence the idea of having different queues for different types of messages. A suggestion here is to give a feasible deadline a whole other notion: a feasible deadline should relate to how much the packet to be delivered is important or more important than other packets. Thus dealing with priorities here could be a good thing. Each packet or message could be given a certain priority or a certain weight. Also, we could think at this point about different classes or different flows of multimedia communication messages. This will most probably lead to a better scheduling when talking about periodic packets. Minimizing energy required to transmit periodic packets over a wireless network is of critical importance, because the wireless network has to handle packets from both periodic and non-periodic sources.

El Gamal, Nair, and Prabhakar [5] considered the problem of scheduling a set of packets with minimum energy. Designing energy efficient transmission policies for randomly arriving traffic with delay constraints has been studied. The MoveRight algorithm has been proposed, which minimizes energy required to transmit packets in a wireless environment. It is motivated by the observation that in many channel coding schemes it is possible to significantly lower the transmission energy by transmitting packets over a longer period of time. Scheduling combined with adaptive power control schemes could potentially yield interesting results. So, the minimum energy scheduling problem for a multiple-user channel (uplink and downlink) involving several transmitters and receivers where time-division is used is investigated. The goal is to minimize the total energy for all users. The setup is as follows: suppose that m packets arrive at the transmitter's buffer in the interval [0,T) at times 0 = t1,t2,t3,...,tm < T. The node is required to transmit all m packets within the interval [0,T]. The question is, how should the packet transmissions be scheduled to minimize the total energy required to transmit the packets?

The packets here can have different energy functions. Also, energy functions are convex and decreasing in the transmission duration, and this is essentially all that is assumed about the channel, transmitters, and receivers. By exploiting the special features of the problem, an algorithm, MoveRight, which finds the global optimal schedule efficiently, is developed. MoveRight iteratively moves the start times of packet transmissions one at a time, so that each move locally optimizes the energy function. The algorithm was shown to solve other scheduling problems, such as when packets have individual deadlines, and when the transmit buffer is finite. MoveRight also leads to an online algorithm that uses a simple look-ahead buffer. The transmitter buffers the packets for a specified length of time (the look-ahead window). At the end of the look-ahead window, the packets in the buffer are scheduled using a faster version of the MoveRight algorithm for transmission from L to 2L (where [0, L] is the interval for arriving packets and L « T). Meanwhile, the arrivals from L to 2L are buffered, to be transmitted in the following time window. Hence, at the expense of incurring a delay of $\approx$ L, packets are scheduled optimally. A window of 20 to 30 time units was shown to be a good choice for L, in the sense that most of the achievable decrease in energy is obtained.

One of the advantages of this algorithm is that it is derived for a multiuser network and it is shown to con-verge to the optimal schedule. Another advantage is that the setting is realistic and thus it is feasible.The MoveRight algorithm, however, is complicated, and requires knowledge of all users' channels as well as their queue lengths. The main disadvantage related to multimedia communication is the observation the paper started with. The whole algorithm is based on the fact that energy will be minimized only if the transmission duration is expanded in time. This has proven to have optimal results when applied in the form of the MoveRight algorithm. However, in multimedia and real-time messages, delay is a very important issue. Real-time messages in particular cannot tolerate delay and thus the algorithm seems far-fetched in wireless networks with delay-intolerant applications. Since multimedia applications are the event of the century, one should think about an alternate to the algorithm with delay minimized. A suggestion is that the MoveRight algorithm should move less right and more linearly straight towards other passes. However this suggestion might be infeasible, more could be said about this issue when more is read about the topic.

Liu, Elhanany, and QiA [6], presented a MAC protocol (Q-MAC) with the advantage of minimizing energy consumption while increasing QoS support in multihop wireless sensor networks. The protocol is based on priority levels where network services are differentiated by their application priority and/or the state of the system resources (residual energy, queue occupancy). The Q-MAC protocol has the distinguishable feature of allowing WSNs to reflect on the criticality of data packets originating from the different sensor nodes. Q-MAC involves both intra-node and inter-node scheduling to provide differentiated services while decreasing energy consumption. The intra-node scheduling scheme adopts a multiple first-in-first-out (FIFO) queuing system to classify data packets according to their application and MAC layer abstraction. A received packet is classified based on its criticality and then stored into the appropriate queue. In Q-MAC, they append five extra bits of information to every message, two for identification of the types of ap-plications and three for the types of sensing data. Also, packets that have gone through more hops have a higher priority. The queuing architecture in Q-MAC consists of five queues with one specified as an instant queue (any packet stored in this queue will be instantly served). The MAX-MIN fairness algorithm [4] and the packetized GPS algorithm [7] are used to determine the next packet to be served from the other queues (the four remaining queues). The inter-node scheduling employs the power conservation MACAW protocol (PC-MACAW) and the loosely prioritized random access protocol (LPRA) for multiple access of the channel among neighboring sensor nodes. The Power Conservation MACAW (PC-MACAW) is a modified version of MACAW, which conquer the energy consumption problem with the classic method. Since idle listening, collision, communication overhead and overhearing contribute most to energy wastage, the authors aim at a simple and distributed protocol to minimize collision and idle listening. The fairness of data transmission among neighboring nodes is ensured by allowing each node to contend for the channel at an identical starting point. And thus the proposed Loosely Prioritized Random Access protocol (LPRA) uses contention time of each node to regulate the order by which nodes access the channel.

As pointed out earlier, a primary concern in real-time multimedia applications in wireless networks is energy efficiency, since a majority of wireless devices operate on batteries that need to be regularly recharged from a power source. It is thus imperative to find ways to overcome the issue of energy consumption. The paper of [6] proposes a protocol (Q-MAC) that aims at minimizing energy consumption while providing QoS guarantees by differentiating network services based on priority levels. This is considered a first advantage of the proposed protocol over the PARM scheduling architecture proposed in [3]. The PARM scheduling algorithm uses the EDF scheduling algorithm but does not differentiate between packets of different priority levels i.e. all packets are of the same criticality. This was considered as a disadvantage of the PARM algorithm and therefore giving the Q-MAC the honors of proposing such an improvement. As opposed to the scheduling algorithm provided in [5], the Q-MAC protocol does not rely on increasing the transmission time in order to minimize the energy needed for transmission. Q-MAC is suitable for real-time messages that cannot tolerate delay or jitter, while the MoveRight algorithm in [5] seems far-fetched in wireless multimedia networks with delay intolerant applications.

Another advantage of the proposed protocol is the provided fairness. The protocol allows for fairness of data transmission between nodes by allowing each node to contend for the channel at an identical starting point. Fairness was not considered in the previous papers [3][5], and it is an advantage in favor of Q-MAC. Also fairness was not considered important in other MAC protocols like S-MAC [8], for they trade off fairness and latency for more energy savings and this is of course not applicable in multimedia environment especially real-time streaming. Also, collision recovery schemes are used in QMAC which is a good point. Two types of recovery are highlighted: doubling the contention window (CW) size, and setting packets dropping threshold according to applications. In the last case, when the difference between the sensing time and the current time is beyond a predefined threshold, packets are immediately dropped. The paper of [6] however presented some disadvantages. First the authors have proposed an intra-node scheduling scheme based on multiple queues queuing system. They also define five different queues where one is an instant queue and the others are with equal priority. From this point, the paper lacks more details where explanation is needed about the other remaining queues and what are their properties. Furthermore, "the number of queues thus determines the number of network service levels" as the authors proclaim. A very prominent disadvantage where one could think that the authors have chosen a random number of queues, and based their service levels on these queues. The contrary is what should be done here. The service levels are first determined and well characterized and then the queues are defined based on these service levels i.e. a queue for each different service level. And thus the right thing to say is: the number of network service levels determines the number of queues.

A challenge here is to choose the proper number of queues and establish the size of each to queue in order to allow traffic conditioning and shaping. The size of the queues is dependent on the node resources and the expected QoS provisioning. The number of queues, though, is an important issue that should be carefully studied and defined. It was pointed out that the number of queues is determined by the number of service levels. So once the network's different service levels have been defined and well characterized, one can use these service levels to define the needed queues. Different types of networks may have different types of service levels. Also the number of service levels is dependent on the overall interest of certain networks.

4 Proposed Scheduling Technique

In this paper, we propose a scheduling technique for WMSNs. We first classify the different types of delivery modes in WMSNs into three categories: static data deli-very mode, snapshot mode, and multimedia-streaming mode. While data delivery modes are typical of scalar networks, snapshot-type are more critical event-triggered observations obtained in a short period of time. Streaming media content is in contrast generated over larger periods of time and thus may require sustained information delivery. From this point of view, we should keep in mind that the priority of live multimedia streaming is higher than the priority of snapshot frames which is in turn higher than the priority of basic data delivery. Having set the ladder of priorities, we simplify the process of queuing and furthermore the process of scheduling. We then propose the following scenario where we imagine a scheme that can be employed for large-scale WMSNs of the future.

Given some geographical area, multimedia sensors that simultaneously serve multiple applications are deployed. Consequently the sensor network is expected to respond to certain queries. We assume that the network is divided into several clusters. We also assume that each cluster has a cluster-head. Cluster-heads are connected between each other and to a base station also called the sink which is responsible for issuing queries. Since such large-scale sensor network would be expected to serve a significant number of queries simultaneously for several applications, the number of attributes sensed by the network would also be substantial. And thus the number of packets arriving at the Cluster-head from different nodes is considerable which necessitates the presence of a scheduler at the cluster-head. Furthermore, the scheduler should be simple in order not to create a computational burden and hence reduce delay and power overhead. A typical scenario would be as follows: The base station (BS) sends a certain query to the cluster-head. Upon receiving query, the cluster-head will propagate it to specified nodes and wait for these nodes to sense the medium and come back with needed data. The nodes will respond by sending packets of data to the cluster-head. The job of the cluster-head is to schedule these packets coming from different nodes to send them in frames to the BS.

Traditional scheduling techniques include the simple FIFO scheduling, Round-Robin and several other techniques. However these algorithms are very simple and they do not take into consideration priorities between different flows. It gives different weights to flows of different priorities. Packets arriving at the cluster-head are queued, we keep track of the number of packets queued for each node as well as their priorities (which is actually related to the mode the node uses: streaming, snapshot or data mode). Having to do this is actually similar to having a different queue for each node and each packet, upon arrival, is inserted in the corresponding queue. This way we can check the size of the queue to know the number of packets arriving from a certain node. However this is too costly since we do not know the actual number of nodes that can be active in a certain cluster (with active meaning that it is currently sensing or communicating data to the cluster-head). So, it is better to have one queue and we can then keep track of the number of packets belonging to each node by having different counters. However, to make things easier, we will imagine that we have a different conceptual queue for each node. Each queue has a size and a priority as shown in Figure 1.

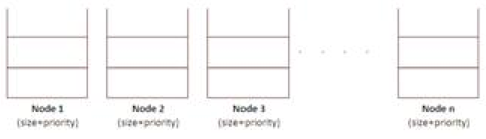


Figure 1: Conceptual Queuing

For the i^th node, we will define the function:

$$F(i) = \frac{s(i) * pr(i)}{\sum_{k=1}^{n} s(k) * pr(k)} \quad (1)$$

where $s(i)$ is the size of the queue of the i^th node, $pr(i)$ is the priority of the node and n is the number of active nodes (i.e. that are currently communicating with the cluster-head). $F(i)$ calculates the percentage of slots given to node i within a certain frame. A frame sent to the BS will contain a certain number of slots; these slots will be divided upon active nodes based on their priorities (depending on the mode it is using: video streaming, snapshots, data delivery) and the strength of the flow between the node and the cluster-head. If two nodes of the same priority have different number of packets enqueued at the cluster-head, the one with more packets should have more slots assigned to it than the other.

An example that illustrates the previous formula: consider a cluster head with four active nodes. The first node has 3 packets available at the cluster and its priority is 1, the second has 2 packets with priority 2, the third has 2 packets with priority 3 and finally the fourth has 3 packets with priority 3. Using equation 1 with n equal to 4 and i ranging between 1 and 4, the percentages will be as follows: the first node will get 10We can calculate the number of slots in a certain frame using the following equation:

$$SPF = \lfloor \frac{size_{frame} - overhead}{size_{slot}} \rfloor \quad (2)$$

where SPF is the number of slots per frame. Note that a set of parameters are network defined, for e.g. the slot size and the frame size. If the frame is large enough, we can set the slot size to be equal to the packet size, this way we eliminate the burden of fragmentation. The $overhead$ are the extra bits added to the beginning of each set of slots that belong to a certain node. These extra bits mainly specify the id of the node to which the slots belong and the number of consecutive slots reserved to this node. A model of the frame is shown in Figure 2.

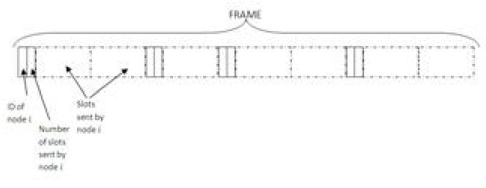


Figure 2: Frame Model

The calculation of the percentage of slots assigned to each node using equation 1 is done in the cluster-head upon receiving a certain query. This means that the cluster-head recalculates the slots assigned to each node when new nodes become active or currently active nodes receive new queries. We assume that, since multimedia streaming might cover eventually long periods of monitoring, the queries will be separated by certain periods of time and thus the recalculation does not impose a burden on the scheduler.

In the next section, we will propose a way to order the slots assigned to a certain node in a way to minimize the jitter between the slots belonging to a certain node in 2 consecutive frames.

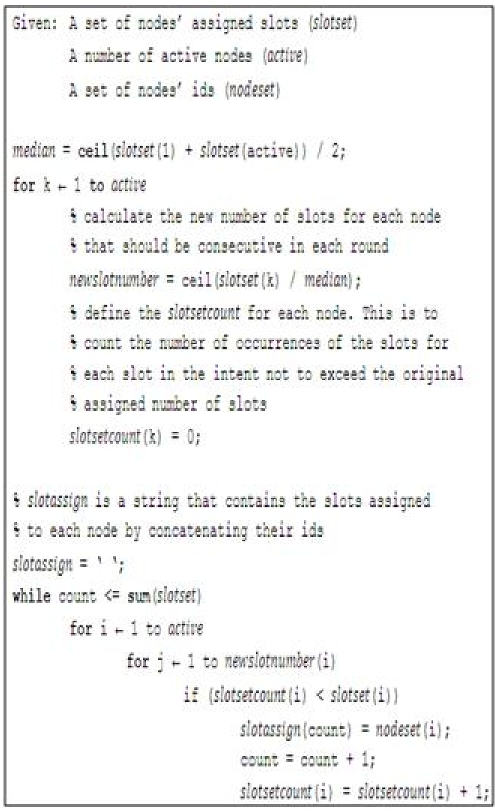
5 Interleaving Slots

As discussed in [2], the order of the slots could have an impact on the maximum jitter between the slots assigned to a certain node. When the cluster-head calculates the number of slots assigned to every node, it can specify an order of the slots. The simplest solution is to put all the slots consecutively also known in the context as the non-interleaving approach. However, a better approach is to interleave the slots to decrease the maximum jitter and delay values. In this paper, we present two algorithms for interleaving the slots.

The first algorithm, given a set of nodes' assigned slot numbers, calculates the median of the set. Then, for each node, the number of slots assigned is divided by the ceiling of the median and the slots are interleaved accordingly. The algorithm is shown in Figure 3.

The second algorithm, instead of interleaving according to the median, uses the smallest in the set to base the interleaving process upon it. The algorithm is shown in Figure 4. It differs from the first algorithm by using the smallest of the set instead of the median. As pointed out in the previous section, overhead is generated in the frame as a result of including extra bits for the id and number of slots assigned to each node. Using one of the previously presented interleaving algorithms, the overhead will obviously increase. Having the divide the consecutive slots for one node into a certain number, say k, and putting each part in a different location in the frame will result in k times more overhead. Each part will now have its own id and number of slots in that partition only.

In the next section, we evaluate the two algorithms for interleaving in terms of minimizing the jitter and we also look at the overhead generated by each. Our candidate algorithm should minimize the jitter but also should not impose a great deal of overhead.

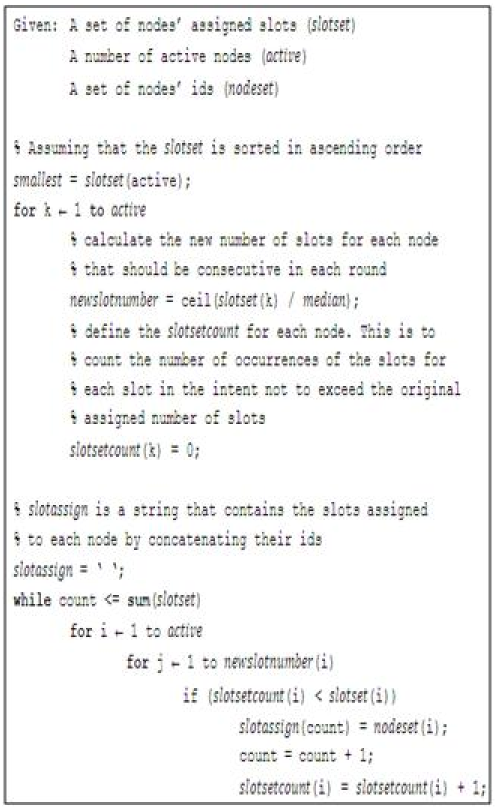


```
Given: A set of nodes' assigned slots (slotset)
       A number of active nodes (active)
       A set of nodes' ids (nodeset)

median = ceil(slotset(1) + slotset(active)) / 2;
for k ← 1 to active
       % calculate the new number of slots for each node
       % that should be consecutive in each round
       newslotnumber = ceil(slotset(k) / median);
       % define the slotsetcount for each node. This is to
       % count the number of occurrences of the slots for
       % each slot in the intent not to exceed the original
       % assigned number of slots
       slotsetcount(k) = 0;

% slotassign is a string that contains the slots assigned
% to each node by concatenating their ids
slotassign = ' ';
while count <= sum(slotset)
       for i ← 1 to active
              for j ← 1 to newslotnumber(i)
                     if (slotsetcount(i) < slotset(i))
                            slotassign(count) = nodeset(i);
                            count = count + 1;
                            slotsetcount(i) = slotsetcount(i) + 1;
```

Figure 3: Interleaving Algorithm 1 (Using Medians)

6 Simulation Results and Analysis

In this section we will first analyze the benefits behind using the interleaving algorithms proposed in the previous section. Secondly, we will compare our scheduling algorithm against four baseline algorithms, namely, MIN-FIFO, MIN-EDF, MAX-FIFO, and MAX-EDF. To compare the interleaving algorithms together and to the non-interleaving approach, we have developed a Matlab program that simulates a set of nodes having assigned slots. We then use the above algorithms to interleave the slots and compare the resulting jitter of different nodes. Since the interleaving of slots will result in more overhead, we also compared the overhead of each of the algorithms. The important parameter here is the number of active nodes in a cluster. We ran the program with a different number of active nodes in each run. Figure 5 shows the average results for 20 active nodes. Obviously, both algorithms minimize the jitter compared to the non-interleaving approach.



```
Given: A set of nodes' assigned slots (slotset)
       A number of active nodes (active)
       A set of nodes' ids (nodeset)

% Assuming that the slotset is sorted in ascending order
smallest = slotset(active);
for k ← 1 to active
       % calculate the new number of slots for each node
       % that should be consecutive in each round
       newslotnumber = ceil(slotset(k) / median);
       % define the slotsetcount for each node. This is to
       % count the number of occurrences of the slots for
       % each slot in the intent not to exceed the original
       % assigned number of slots
       slotsetcount(k) = 0;

% slotassign is a string that contains the slots assigned
% to each node by concatenating their ids
slotassign = ' ';
while count <= sum(slotset)
       for i ← 1 to active
              for j ← 1 to newslotnumber(i)
                     if (slotsetcount(i) < slotset(i))
                            slotassign(count) = nodeset(i);
                            count = count + 1;
                            slotsetcount(i) = slotsetcount(i) + 1;
```

Figure 4: Interleaving Algorithm 2 (Using Smallest)

Obviously, both algorithms minimize the jitter compared to the non-interleaving approach. The first algorithm (using the median) shows a significant decrease in the jitter for the nodes with highest number of slots assigned to them. However, for the nodes with least number of slots assigned, we can notice a multiplicative increase. The second algorithm (using the min) does not do as well as the first but it shows consistency in the results. The jitter is almost the same for all nodes. In both cases the jitter is drastically decreased. The evaluation does not stop here. We know that the more the interleaving of slots of the same node in one frame, the more overhead is generated. For this reason, we developed another Matlab program that compares the overhead generated from both algorithms and we compared to non-interleaving the slots. The program calculates the overhead as a function of the number of active nodes. The results are shown in Figure 6 where the overhead is calculated for up to 50 active nodes. As seen in the figure, the first algorithm performs the worst. Non-interleaving obviously does not generate that much of overhead since all slots of the same node are clustered together. This means that the overhead needed is a multiple of the number of active nodes. The second algorithm, although it imposes some overhead, however the overhead is affordable compared to the first algorithm. We can conclude that the second algorithm is better since it minimizes the jitter while keeping the overhead up to a certain limit.

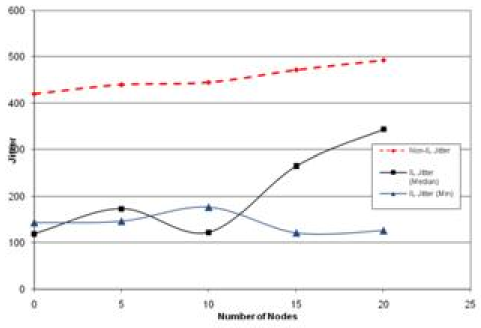


Figure 5: Interleaving compared with 20 active nodes

The second part of our simulation analysis is focused on the scheduling algorithm.To quantitatively evaluate our scheduling algorithm, we compared our algorithm against the following four baseline algorithms, namely, MIN-FIFO, MIN-EDF, MAX-FIFO, and MAX-EDF.

- MIN-FIFO: Admitted messages are transmitted based on the First-In-First-Out policy. For each admitted message, the MIN-FIFO algorithm assigns the lowest bandwidth (e.g. 125 kbps), which leads to the lowest energy consumption.

- MIN-EDF: Messages admitted by the system are delivered using the EDF policy. Similar to the MIN-FIFO algorithm, MIN-EDF transmits each admitted message using the lowest bandwidth.

- MAX-FIFO: The First-In-First-Out policy is employed to schedule messages, and the highest bandwidth (e.g. 1000 kbps) is assigned to all admitted messages.

- MAX-EDF: The EDF policy is used to schedule all admitted realtime messages. Regarding transmission rate assignment, MAX-EDF allocates the highest bandwidth to admitted messages.

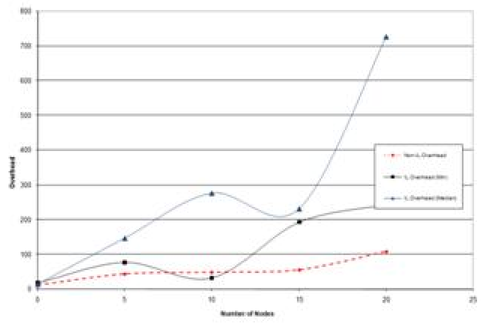


Figure 6: Overhead generated by interleaving

The performance metrics that we used to evaluate and compare the scheduling algorithms are:

1. Total Power Consumption: The power consumed as a result of processing, transmitting and receiving a message based on MICA motes.

2. Missed Rate: Fraction of the total submitted messages that missed their deadline.

The total power consumption of the MAX-FIFO, MAX-EDF, and our algorithm is analyzed in Figure 7. The deadline base is increased from 100 to 500 Seconds. Our algorithm is much more energy-efficient than MAX-FIFO and MAX-EDF. We attribute the performance improvement to the fact that our scheme noticeably reduces energy consumption by judiciously assigning transmission rate for each admitted messages.

The deadlines are randomly generated in the range between 100 to 500 seconds, and the data sizes are randomly chosen in the range from 500 to 1000 Kbytes. We observe from Figure 8 that our algorithm performs better than MIN-FIFO and MIN-EDF in terms of missed rate over. This is mainly because MIN-FIFO and MIN-EDF assign the minimal transmission rate to admitted messages, resulting the maximal transmission time.

7 Conclusion and Future Work

In this paper, we have presented a novel scheduling algorithm for WMSNs. It divides the frame sent from the cluster-head to the BS into slots and gives a percentage of these slots into each node. These percentages are calculated based on a formula that we came up with. The formula takes into consideration the priority of the node and the strength of the flow from that node into the cluster-head. This formula is recalculated each time the cluster-head receives a new query with new nodes involved. Since in WMSNs the multimedia streaming might cover eventually long periods of monitoring, the queries will be separated by certain periods of time and thus the recalculation does not impose a burden on the scheduler. We also propose an interleaving scheme for the slots in a certain frame. We first came up with two algorithms to compare their efficiency. We also compared these algorithms to the non-interleaving approach and we found that the algorithm that divides by the smallest of the set is the most consistent in minimizing jitter and the most efficient in keeping the overhead bounded by a certain limit.

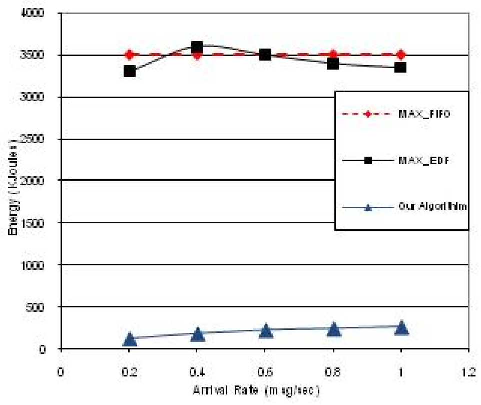


Figure 7: Total power consumed as a function of the arrival rate

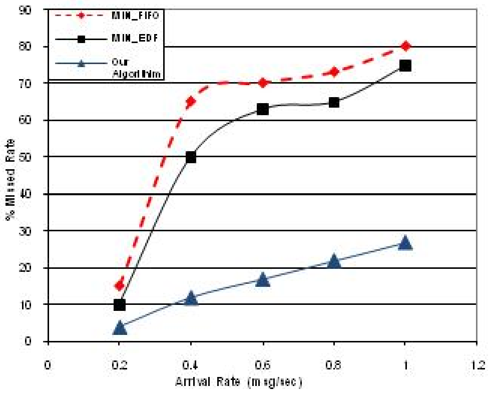


Figure 8: Missed rate as a function of the arrival rate

References

[1] Akyildiz, I. F., Melodia, T., and Chowdhury, K. R. A survey on wireless multimedia sensor networks. *Computer Networks*, 51:921–960, 2007.

[2] Alexander, S., Olli, A., Juha, K., and Timo, H. Ensuring the qos requirements in 802.16 scheduling. In *MSWiM '06: Proceedings of the 9th ACM international symposium on Modeling analysis and simulation of wireless and mobile systems*, pages 108–117, New York, NY, USA, 2006. ACM.

[3] Alghamdi, M. I., Xie, T., and Qin, X. Parm: A power-aware message scheduling algorithm for real-time wireless networks. In *Proc. of ACM Workshop on Wireless Multimedia Networking and Performance Modeling (WMuNeP*, pages 86–92. ACM Press, 2005.

[4] Gallager, R. and Bertsekas, D. *Data Networks*. Prentice Hall India, 2nd edition, 1987.

[5] Gamal, A. E., Nair, R., Prabhakar, B., Uysal-biyikoglu, E., and Zahedi, S. Energy-efficient scheduling of packet transmissions over wireless networks. In *In Proceedings of IEEE Infocom*, pages 1773–1782, 2002.

[6] Liu, Y., Elhanany, I., and Qi, H. An energy-efficient qos-aware media accesscontrol protocol for wireless sensor networks. In *in Proceedings of the IEEE International Conference on Mobile Adhoc and Sensor Systems*, 2005.

[7] Parekh, A. K. and Gallager, R. G. A generalized processor sharing approach to flow control in integrated services networks: The single-node case. *IEEE/ACM Transactions on Networking*, 1:344–357, 1993.

[8] Ye, W., Heidemann, J., and Estrin, D. Medium access control with coordinated adaptive sleeping for wireless sensor networks. *IEEE/ACM Transactions on Networking*, 12:493–506, 2004.