Intertwined Medium Access Scheduling of Upstream and Downstream Traffic in Wireless Sensor Networks

Miloud Bagaa*, Mohamed Younis§, Djamel Djenouri* and Nadjib Badache*
* CERIST Research Center, Algiers, Algeria. Emails:{bagaa,ddjenouri,badache}@mail.cerist.dz
§ Dep. of Comp Sc. & Elect Eng, Univ. of Maryland Baltimore County. Email:younis@cs.umbc.edu

***Abstract*—In wireless sensor networks, the sensor data are often aggregated en-route to the base-station in order to eliminate redundancy and conserve the network resources. The base-station not only acts as a destination for the upstream data traffic, but it also configures the network by transmitting commands downstream to nodes. The data delivery latency is a critical performance metric in time-sensitive applications and is considered by a number of data aggregation schemes in the literature. However, to the best of our knowledge, no solution has considered the scheduling of downstream packets, originated from the base-station, in conjunction with upstream data aggregation traffic. This paper fills such a gap and proposes *MASAUD*, which intertwines the medium access schedule of upstream and downstream traffic in order to reuse time slots in a non-conflicting manner and reduce delay. *MASAUD* can be integrated with any scheme for data aggregation scheduling. The simulation confirms the effectiveness of *MASAUD*.**

***Index Terms*—wireless sensor network, data aggregation, medium access scheduling.**

I. Introduction

A wireless sensor network (*WSN*) is typically composed of many sensor nodes deployed in an area of interest. Each sensor node is able to measure ambient conditions and monitor its environment. It is also equipped with a limited energy supply and a short range half-duplex radio transceiver. The data collected by the sensors are disseminated to an in-situ base-station that correlates the readings of the various sensors for an overall situational awareness. Due to the large deployment area and the short communication range, and to save energy, the data packets are forwarded to the base-station over multi-hop paths. The base-station not only acts as a sink of all data traffic but also it interfaces the *WSN* to remote command centers, and queries, tasks, configures and synchronizes the sensor nodes. *WSNs* have numerous applications, such as detecting forest fire and monitor dangerous biological and chemical agents. In this type of applications, it is very critical to detect and deliver alarms about serious events in a timely manner. The base-station should also quickly respond with commands to configure the sensor nodes as appropriate for emerging events.

In-network data aggregation is a widely-used ~~as~~ optimization strategy in *WSN*. The goal is to reduce energy consumption and increase the communication bandwidth by eliminating redundant data packets en-route to the base-station. Prior to forwarding a data packet, an aggregation node should wait for receiving data from all its downstream predecessor on the routing tree. This may increase data delivery latency and have a negative impact on event notification in *WSN*. One of the most popular schemes that are used for reducing latency is to limit of medium access collisions by adopting reservation based protocols such as *TDMA* (Time Division Multiple Access). Recently cross-layer methods have been proposed to combine the support of data aggregation with medium access scheduling in order to minimize the data delivery delay and cut on the energy consumption [1]–[3]. In these solutions a tree is created and slots are assigned to nodes such that the data are gathered and aggregated en-route to the base-station while avoiding competition for medium access among the network nodes. To observe the order of transmissions on the aggregation tree, ascending time slots are to be assigned based on the child-parent relationship on the tree.

A key limitation of existing solutions is that they do not account for the communication from the base-station to the sensor nodes. Basically, it has been assumed that the base-station is able to reach all sensors through a single broadcast [4]. However, this is not often a practical assumption. The *WSN* may be deployed in an area with rough terrain and the base-station may not have a line-of-sight link to all sensors; thus the base-station would only be able to reach the sensors over multi-hop routes. Furthermore, in security-related applications such as combat field surveillance the base-station's identity and location should not be revealed to adversaries; therefore, the base-station would avoid making a long range transmission that can be intercepted and tracked in order to determine the position of the base-station. In summary, the base-station could be forced to send commends and queries to sensors over multi-hop links, which makes it necessary to schedule both upstream and downstream transmissions in the *WSN*. Existing solutions do not factor the packets sent by the base-station and would deal with the upstream and downstream traffic independently, which will extend the frame size and increase latency.

In this paper, we present *MASAUD*, a novel cross-layer approach for Medium Access Scheduling of Aggregated Upstream and Downstream traffic in *WSN*. The main objective of *MASAUD* is to enable collision-free collection of aggregated data and supporting base-station-to-sensors communication while reducing time latency. *MASAUD* can be executed in conjunction with any data aggregation scheduling solution. In other words, *MASAUD* aims to minimize the time latency needed to support downstream traffic from the base-station while taking into account any existing data aggregation schedule. In order to achieve such design goal, *MASAUD* intertwines the transmission of upstream and downstream packets in order to increase time slot reuse. *MASAUD* not only reduces latency but also boosts the network throughput and improves utilization of the wireless channel. The effectiveness of *MASAUD* is validated via simulation. To the best of our knowledge this is the first paper that considers the scheduling of both aggregated sensor data traffic and base-station command/control broadcasts in the *WSN*.

The related work is summarized in the next section. Section III analyses the problem of scheduling downstream traffic in WSNs. *MASAUD* is presented in section IV. The simulation results are discussed in Section V. Finally, Section VI concludes the paper.

II. Related work

The data aggregation scheduling problem has been extensively studied in the literature. Published algorithms typically pursue a two-phase process. In the first phase, each node is assigned a rank based on breadth first ordering relative to the base-station, i.e., the rank reflects the number of hops on the shortest path from a node to the base-station. The node ranks are used to construct the data aggregation tree. The second phase sets a transmission schedule based on the data aggregation order in the tree. The main goal of most existing solutions is to maximize the network throughput by gathering the data from all network nodes in the least time. To ensure that the aggregated data are collected in the same time frame, the slot allocated to a parent ($rank = j$) is greater than all the ones of its children ($rank = j + 1$) on the aggregation tree.

Chen et al. [1] are the first to consider the problem of data aggregation scheduling. They proved that the problem is NP-hard and proposed an heuristic applied on the shortest path tree. Huang et al. [2] proposed a centralized solution called *NCA*. The network radius is defined as the maximum number of hops between the network center and any other node. *NCA* firstly constructs a Connected Dominating Set (*CDS*) and then schedules the network nodes by adopting first-fit allocation. An enhancement of *NCA* is proposed in [5]. On the other hand, *ACS* [3] uses a Balanced Shortest Path Tree instead of a *CDS*. Unlike all the centralized algorithms discussed above, Yu et al. [6] proposed a distributed solution, called *DAS* was later enhanced by Xu et al. [7]. Bagaa et al. proposed a centralized solution in [8] in which no information about candidate parents or children are provided to the scheduling algorithm. The tree construction and the scheduling are performed concurrently. However, all these solutions do not take into account the communication from the base-station to the network nodes.

On the other hand, some work considered the communication from and to the base-station. In [9], the base-station periodically broadcasts interest messages, referred to as request diffusion, to notify the sensor nodes about what to report. During the dissemination of interest messages throughout the network, a tree is formed to support the routing of sensor data to the base-station. IRIS [10] implements some improvement to reduce the time latency and the number of medium access collisions. Basically, the request diffusion is probabilistically disseminated rather than flooded. Meanwhile, *DRIP* [11] disseminates the commands and requests to a limited set of nodes instead of all the network nodes. The implementation of *DRIP* is available in the official distribution of TinyOS [12]. While contention-based *MAC* is assumed in [9]–[11], a few of protocols [13] [14] support communication from the base-station to the sensor nodes using TDMA mechanism. In contrast to these solutions, *MASAUD* takes into account both communication from and to the base-station by exploiting *TDMA* scheme. In *MASAUD*, the slots of upstream and downstream traffic are intertwined in order to reduce the time latency.

III. Problem Formulation

We assume that all sensor nodes have the same transmission range, say r, and have synchronized clocks. Each node has a single half-duplex radio transceiver, i.e., it cannot transmit and receive simultaneously and cannot hear two messages at the same time. Time-based medium access arbitration is assumed. Although, the communication range "ς" of the base-station is larger than that of a sensor, i.e., $\varsigma > r$, the base-station may not be able or allowed to reach all sensors within a circle of radius ς due to line-of-sight limitations or security concerns, as stated in Section I. We model a *WSN* as a graph $G = (V, E)$, where V is the set of all nodes in the network, with a base-station $B \in V$, and E is the set of edges among the nodes in G. We assume symmetric links, and thus an edge $(u, v) \in E$ implies that the two nodes, u and v, are within the transmission range of each other. In addition to collecting the data of all sensors, we assume that the base-station manages the network operation by setting the routing topology and medium access schedule for all nodes. In other words, *MASAUD* is to be executed by the base-station.

Assume that $\Omega = (\Gamma, T)$ is a time-based medium access schedule for routing aggregated data over a multi-hop aggregation tree, where T is the set of time slots in which the sensors transmit their data and $\Gamma = V - \{B\}$. Given $\Omega = (\Gamma, T)$, the problem which we tackle is how to augment such medium access schedule so that each command/request can be forwarded from the base-station to all sensor nodes without collisions in the least time. The command schedule can be defined as a sequence of sender sets $S_i, S_j, \cdots, S_l$, where $i < j \cdots < l$. The base-station transmits in a slot $q < i$ to nodes in S_i. The nodes in S_i forward the command without collision to the nodes in S_j in time slot i, those in S_j retransmit the command in time slot j, and so on. In the last time slot l, the command should be received by all the network nodes. The Minimum Latency Command Scheduling (*MLCS*) problem is to find a schedule $q, i, j, \cdots, l$ and the sets $\{S_i, S_j, \cdots, S_l\}$, such that there is no medium access conflict with the data aggregation schedule and the value of l, which reflects the delivery latency to all network nodes, is minimized.

An instance of the *MLCS* problem must satisfy the following conditions:

1) $S_i \cap S_j = \emptyset, \forall i \neq j$;
2) A command should be received to all network nodes at or before time slot l. The commands are forwarded over multi-cast tree (V, E_T) rooted at B, such that $E_T \subseteq E$ represents the multi-cast tree edges.
3) *Collision-free constraint*:
   - u can be scheduled at time slot τ if the following conditions hold:
     - There is not any neighbor of u that receives a packet, at time slot τ in $\Omega = (\Gamma, T)$;
     - There is not any child of u, in the multi-cast tree, that sends a packet at time slot τ in $\Omega = (\Gamma, T)$;
     - There is not any neighbor of u's children, in the multi-cast tree, that sends a packet, at time slot τ in $\Omega = (\Gamma, T)$;
   - Two nodes u and v cannot be scheduled at the same time slot iff: there is a child of u neighbor to v or a child of v neighbor to u.
4) *Transmission order constraint*: To meet the freshness requirement in sensor networks, meaning command should received by all sensor nodes in the same time frame, the time slot at which each child node in the multi-cast tree transmits command must be larger than that of its parent.

IV. Medium Access Scheduling of Upstream-Downstream traffic

In order to reduce the time latency the number of broadcast should be reduced in the network. For this reason, *MASAUD* adopts a broadcast tree to disseminate the commands from the base-station to all the network nodes. Moreover, to reduce the time latency, in *MASAUD* the base-station can adjust its communication range, by varying its transmitting power, to reach more or fewer sensors in one transmission. Thus, it makes sense to assume that the transmission range equals $k \times r$, where $k = 1, 2, \cdots \lfloor \frac{S}{r} \rfloor$. Therefore, the multi-cast tree will become hop by hop after the k hops from the base-station. For the sake of simplicity, during the explanation of *MASAUD*, we consider that the base-station sets $k = 1$ and transmits at the same range of a sensor node.

MASAUD is executed in two steps. First a multi-cast tree is formed, then the non-leaf nodes in the multi-cast tree are scheduled. During the scheduling process, we consider that the data aggregation schedule is already established. Thus, the command schedule should take into account the existing data aggregation schedule. In order to facilitate the presentation of *MASAUD*, the data aggregation schedule depicted in Fig. $1(a)$ is considered as a working example throughout this section. In this figure, the depth of the breadth-first tree is four, i.e., 4 levels. An arrow between two nodes, a and b, indicates that a has chosen b as its parent. The dotted lines represent the graph connectivity. The number besides the solid arrow (a, b) represents a's transmission slot. The time latency of data aggregation scheduling in this example is 7 time slots. Fig.

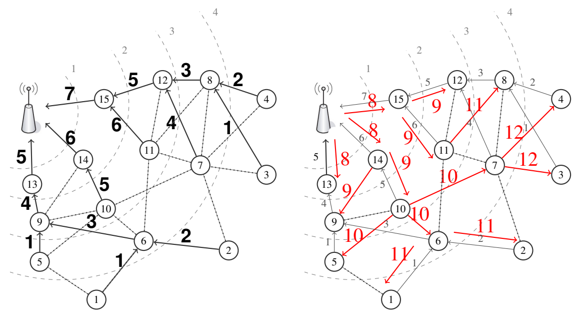

(a) (b)

Fig. 1: Scheduling of upstream and downstream traffic: intuitive solution

$1(b)$ shows the execution of an intuitive solution to schedule downstream traffic for tree in Fig. $1(a)$. In such a solution, the downstream schedule starts after the base-station receives the data, i.e., in 8 slots. The time latency achieved by the intuitive solution in this example is 12 time slots. Recall that the data aggregation tree and the medium access schedule are considered as input for MASAUD algorithm. During the explanation of MASAUD, Fig. 2 will be referenced throughout the discussion.

A. Multi-cast tree construction

We assume that the network nodes are already organized into levels, such that the nodes which have the same hop-count to the base-station, in breadth-first ordering belong to the same level. As depicted in Fig $1(a)$ nodes 1, 2, 3 and 4 belong to level four, whereas nodes 5, 6, 7 and 8 are at level three. The multi-cast tree would behave as virtual backbone of network nodes. The proposed tree in this paper is like the one in [7] with some differences. The proposed tree in [7] is converge-cast, whereas the proposed tree here is a multi-cast. In contrast to the converge-cast tee, the multi-cast tree is constructed in a way that reduces the time latency when the commands are forwarded from the base-station to sensor nodes. Forming the multi-cast tree follows three steps:

1) Identifying Maximum Independent Set (MIS): MIS is constructed first by selecting a set of nodes that do not have links between them in G. Let *neighbors_MIS* denotes the set of nodes which are neighbors to the nodes in *MIS*. Initially, *MIS* include the base-station and *neighbors_MIS* will have the neighbors of the base-station. Afterward, for each node u in level L, starting from the first level, the following test is done: If $u \notin neighbors\_MIS$, u will be added to MIS and its neighbors to *neighbors_MIS*. At the end, the MIS is found as depicted in Fig. $2(a)$. The nodes in MIS are called dominator nodes, whereas the others are called dominatee nodes. In the figure, the dominator nodes in *MIS* are presented with black circles, whereas the dominatee nodes are presented with white circles.

2) Connected Dominating Set (CDS) construction: CDS is constructed by interconnecting the nodes in *MIS* together using a set of non-redundant dominatee nodes (called connectors). Here, a connector node x (a dominatee of a dominator u) is said to be non-redundant for the dominator u, if removing

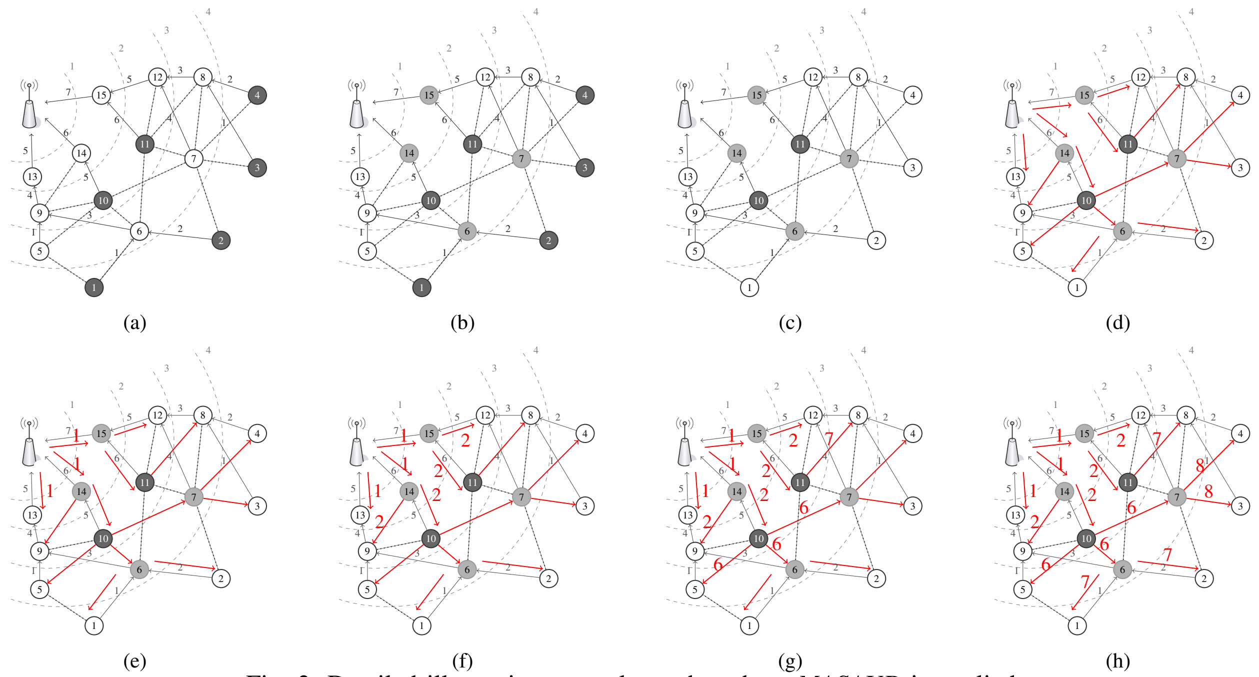


Fig. 2: Detailed illustrative example to show how *MASAUD* is applied

Algorithm 1 MASAUD: Command and requests scheduling

```
Input:
    $Levels = \{L_0 \cdots L_R\}$: network nodes ($V$) organized into levels,
                                  such that $B \in L_0$.
    $Nodes = \cup_{i=0}^{R} L_i$: $V$.
Output:
    $TC$: The time slots assigned to nodes to broadcast a command or request
  in the multi-cast tree.
1:  for all $u \in Nodes$ do
2:      $TP(u) = 0$;
3:      $\mathcal{F}(u) = \{TD(u)\}$;
4:      for all $v \in \gamma(u)$ do
5:          $\mathcal{F}(u) = \mathcal{F}(u) \cup RD(v)$;
6:          if $v \in Child_C(u)$ then
7:              $\mathcal{F}(u) = \mathcal{F}(u) \cup TD(v)$;
8:          end if
9:      end for
10: end for
11: for all $L \in Levels$ do
12:     for all $u \in L$ do
13:         if $u \in CDS$ then
14:             $TC(u) = TP(u) + 1$;
15:             for all $TC(u) \in \mathcal{F}(u)$ do
16:                 $TC(u) = TC(u) + 1$;
17:             end for
                // $u$ is scheduled at $TC(u)$
18:             for all $v \in Child_C(u)$ do
19:                 $TP(v) = TC(u)$;
20:                 for all $w \in \gamma(v)$ do
21:                     $\mathcal{F}(w) = \mathcal{F}(w) \cup TC(u)$;
22:                 end for
23:             end for
24:         end if
25:     end for
26: end for
```

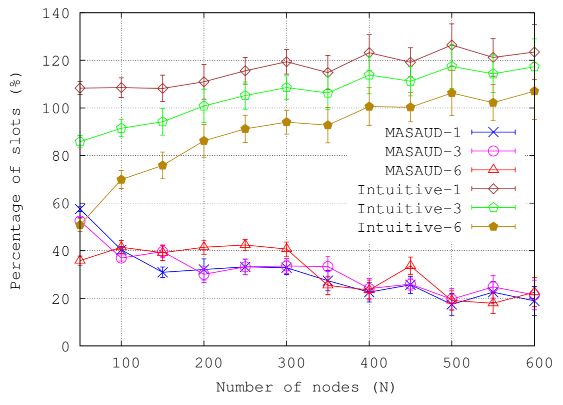
x will disconnect at least one of the two-hop dominator neighbors of u. Starting from the base-station ($level = 0$), the dominator nodes in level L would be interconnected with the other ones in level $L+1$ and $L+2$ by using the connectors in level $L+1$. To select the connectors which enable more reduction of the time latency, the dominatee nodes in level $L+1$ are sorted according to the number of their neighbors, where the node with the highest number is first considered. Let S denote the set of sorted dominatee nodes in level $L+1$. The connectors are selected from S, such that all the dominator nodes in levels $L+1$ and $L+2$ are connected with the ones in level L. The remaining nodes are denoted as dominatee nodes. As depicted in Fig. 2(b), the *CDS* is constructed. In the figure, the black, gray and white circles represent the dominator, connector and dominatee nodes, respectively.

3) Multi-cast tree construction: The multi-cast tree is constructed by doing two steps: Firstly, the dominator nodes, which are leaves, are moved from the dominator to domminatee set as depicted in Fig. 2(c). The leaf nodes are removed from *CDS* as there is no need to forward a command from these nodes. Then, a tree is constructed level by level starting from the base-station as follows: (a) Each dominator node in level L selects its children from the connectors in level $L+1$; (b) The connectors in level L select their children from the dominators in levels L and $L+1$; (c) The remaining nodes, i.e., domminatees, in level L select their parents from the dominators and connectors in levels L and $L-1$. The formed multi-cast tree is depicted in Fig. 2(d).

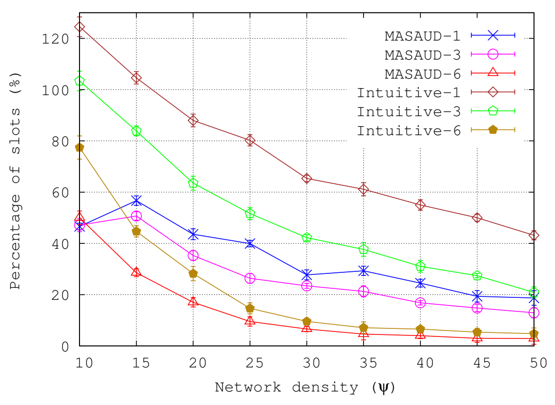
B. Commands and requests scheduling

The following notation will be used through this subsection:

- (u, v): Link between nodes u and v.
- $Child_D(u)$: The children of node u in data aggregation tree. For example in Fig. 2(d), $Child_D(14) = \{10\}$.
- $Child_C(u)$: The children of node u in multi-cast tree, e.g., $Child_C(14) = \{9, 10\}$ in Fig. 2(d).
- $TD(u)$: The time slot assigned to node u in the aggregation tree to transmit its data. For example in Fig. 2(a),



(a)



(b)

Fig. 3: Comparison of the data delivery latency of *MASAUD* and the intuitive solution: applied on DAS-UT

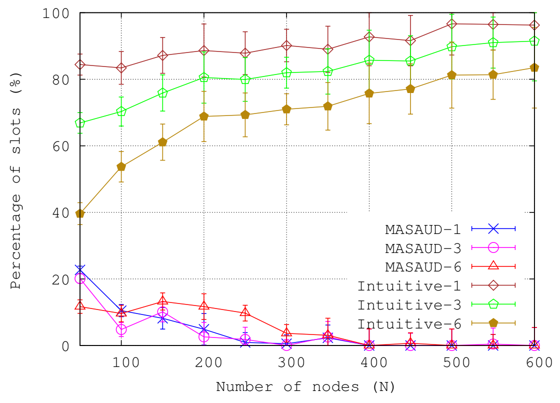
$TD(3) = \#1$ and $TD(4) = \#2$.

- $RD(u)$: Set of time slots assigned to node u in the aggregation tree to receive packets from its children. In Fig. $2(a)$, $RD(9) = \{\#1, \#3\}$ and $RD(12) = \{\#3, \#4\}$.
- $TC(u)$: The time slot assigned to node u to broadcast a command or request to its children in the multi-cast tree.
- $\gamma(u)$: Set of u's neighbors in G.
- $\digamma(u)$: Set of time slots that should not be used by node u in order to avoid collisions.
- $TP(u)$: The time slot of u's parent in the command transmission scheduling, and initially is set to zero.
- $\rho(u)$: The parent of node u in the multi-cast tree.

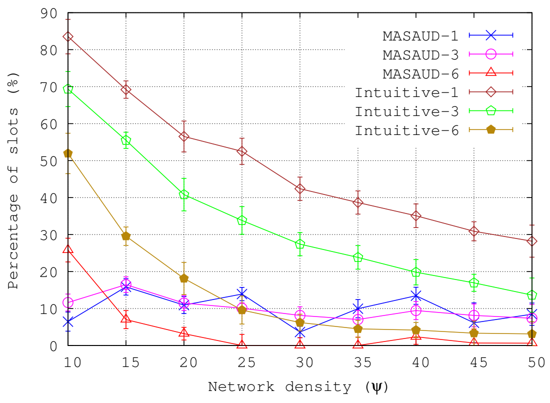
The different steps of scheduling downstream packets are depicted in pseudo-code Algorithm 1. Initially, only the data aggregation tree schedule is considered, and hence $\digamma(u)$ is initialized by all the transmitting slots of $\{u\} \cup Child_C(u)$ (i.e., TD) and all the receiving slots (i.e., RD) of $\gamma(u)$. Formally, $\digamma(u) = \{TD(u)\} \cup \{RD(v), \forall v \in \gamma(u)\} \cup \{TD(v), \forall v \in Child_C(u)\}$ (Algorithm 1: Lines $1-10$). As depicted in Fig. $2(d)$, $\digamma(B) = \{\#4, \#5, \#6\} \cup \{\#5, \#6, \#7\} = \{\#4, \#5, \#6, \#7\}$, whereas $\digamma(11) = \{\#1, \#2, \#3, \#4, \#6\}$. Since the commands transmission schedule start at the base-station, it should be assigned the smallest collision-free slot. In other word, $TC(B)$ should be the smallest time slot which is not in $\digamma(B)$. Since $\digamma(B) = \{\#4, \#5, \#6, \#7\}$, the first time-slot that can be assigned to the base-station is $\#1$, i.e., $TC(B) = \#1$, as depicted in Fig. $2(e)$. *MASAUD* then processes the remaining levels, one by one in ascending order. For each dominator or connector node u, in level L, *MASAUD* does the following. Firstly $TC(u)$ is set to be the smallest time slot which is higher than $TP(u)$ and does not belong to $\digamma(u)$ (Algorithm 1: Lines $14-17$). For example in Fig. $2(f)$, $\rho(14) = B$, $TP(14) = \#1$ and $\digamma(14) = \{\#1, \#3, \#5, \#6\}$, and thus the time slot (i.e., $TC(14)$) which can be assigned to node 14 is #2. After scheduling u, all $\digamma$ of the neighbors of u's children, in the multi-cast tree, are updated by adding $TC(u)$. Formally, $\forall v \in Child_C(u), \forall w \in \gamma(v) : \digamma(w) = \digamma(w) \cup \{TC(u)\}$ (Algorithm 1: Lines $18-23$). To illustrate, let us consider the example in Fig. $2(f)$. Before the scheduling of node 11, $\digamma(7) = \{\#1, \#2, \#3, \#4\}$. When node 11 is scheduled at time slot #7, the set $\digamma$ of time slots to be avoided is updated for each neighbor of 11's children. As node 7 is a neighbor of node 8 (which is a child of node 11), $\digamma(7)$ is updated by adding the time slot #7. For this reason, as shown in Fig. $2(h)$, node 7 cannot use the time slot #7. Using *MASAUD*, the command/request broadcast is intertwined with the data aggregation schedule. As illustrated by Fig. $2(h)$, *MASAUD* could achieve that by adding only one time slot to the data aggregation schedule (total of 8 slots), in contrast to intuitive solution shown in Fig. $1(b)$, which needs frame of 12 time slots.

V. Simulation Results

MASAUD is validated through the simulation. To the best of our knowledge, this paper is the first to tackle the problem of downstream and upstream traffic scheduling in *WSN*. For this reason, the performances of *MASAUD* would be compared only to the intuitive solution mentioned in Section IV. *MASAUD* is evaluated in terms of time latency which is defined as the percentage of time slots increase to permit the base-station to receive the aggregated data and for its transmitted command packet to reach all the sensor nodes relative to the original schedule where only the upstream traffic is scheduled. Formally, the percentage of time slots increase P can be defined as follows: $P = \frac{l}{l'} \times 100$, where l' denotes the number of time slots required to handle only the upstream traffic and l the number of time slots required to handle the downstream and upstream traffic. The choice of data aggregation scheduling solution has an impact on *MASAUD* performances. For this reason, in this simulation the following two data aggregation scheduling solutions are considered: (i) *DASUT* [8], which is centralized; (ii) *DAS* [6], which is distributed. To show the impact of base-station transmission range on the time latency three cases are considered: (i) The base-station sets its range ς to r like the sensor nodes, i.e., it can reach the nodes belonging to the first level of aggregation tree; (ii) The base-station reaches the nodes belonging to the first and second level using only one hop broadcast (i.e., $\varsigma = 2 \times r$); (iii) The nodes in levels $1-6$ are covered by a single transmission, i.e., $\varsigma = 6 \times r$. In the simulation



(a)



(b)

Fig. 4: Comparison of the data delivery latency of *MASAUD* and the intuitive solution: applied on DAS

experiments, N nodes are deployed according to a uniform random distribution. The performance evaluation is performed by varying the number of nodes N and network density ψ. We conduct two types of experiments: (i) we vary N and fix ψ to 5; (ii) we vary ψ and fix N to 300. In our simulation results, each plotted point represents the average of 10 executions. We plot the 95% confidence interval in the graphs.

Fig 3 shows the performances of *MASAUD* and the intuitive solution when applied in conjunction with *DASUT*, whereas, Fig. 4 shows their performances when *DAS* is used as underlying data aggregation scheduler. Fig 3(a) and 4(a) show that whatever the number of nodes and the transmission range of the base-station, the performances of *MASAUD* is almost 100% better than the intuitive solution. We recall that *DASUT* and *DAS* ensure only the data aggregation scheduling. From Fig. 3 and Fig. 4, *DASUT* has better performances than *DAS*, which is which is consistent with published results; therefore, more time slots could be exploited in *DAS* than *DASUT*. Fig 3(b) and 4(b) show that whatever the network density, *MASAUD* significantly outperforms the the intuitive solution. We can also observe in Fig. 3 and 4 that the effectiveness of *MASAUD* is sustained as the number of nodes and network density grow; meaning that the added slots to the frame stay almost constant as shown in Figure 4. Overall the simulation results confirms the effectiveness of *MASAUD*.

VI. Conclusion

In time-sensitive applications of wireless sensor networks (*WSNs*), it is very critical to quickly detect and deliver alarms about serious events. In-network data aggregation scheduling solutions are proposed to reduce the time latency of upstream traffic, i.e., packets originated from the sensor nodes toward the base-station. However, these solutions do not account for downstream traffic generated by the base-station to task, configure and query the sensor nodes both proactively and in response to the detected events. In this paper we have presented *MASAUD* to overcome this shortcoming. *MASAUD* opts to schedule the dissemination of the broadcast traffic sent by the base-station while minimizing latency. The idea is assign time slots that do not interfere with the upstream traffic of aggregated data. *MASAUD* is the first protocol which intertwines upstream and downstream traffic scheduling in *WSNs* in order to reduce delay. *MASAUD* can be integrated with any scheme for data aggregation scheduling. The simulation results have demonstrated the effectiveness of *MASAUD* in achieving low latency.

Acknowledgment: Dr. Younis' work is supported by the National Science Foundation, award # CNS 1018171.

References

[1] X. Chen, X. Hu, and J. Zhu, "Minimum data aggregation time problem in wireless sensor networks," in *Proc. Springer-Verlag MSN'05*, 2005, pp. 133–142.

[2] S. Huang *et al.*, "Nearly constant approximation for data aggregation scheduling in wireless sensor networks," in *Proc. IEEE INFOCOM'07*, May 2007, pp. 366–372.

[3] B. Malhotra, I. Nikolaidis, and M. Nascimento, "Aggregation convergecast scheduling in wireless sensor networks," *Springer Wireless Networks*, vol. 17, no. 2, pp. 319–335, 2010.

[4] M. Khiati and D. Djenouri, "Bod-leach: broadcasting over duty-cycled radio using leach clustering for delay/power efficient dissimilation in wireless sensor networks," *Wiley International Journal of Communication Systems*, 2013.

[5] P.-J. Wan *et al.*, "Minimum-latency aggregation scheduling in multihop wireless networks," in *Proc. ACM MobiHoc'09*, May 2009, pp. 185–194.

[6] Y. Bo, L. J, and Y. Li, "Distributed data aggregation scheduling in wireless sensor networks," in *Proc. IEEE INFOCOM'09*, April 2009, pp. 2159–2167.

[7] X. H. Xu *et al.*, "An improved approximation algorithm for data aggregation in multi-hop wireless sensor networks," in *Proc. ACM FOWANC09*, New Orleans, Louisiana, USA, May 2005.

[8] M. Bagaa *et al.*, "Semi-structured and unstructured data aggregation scheduling in wireless sensor networks," in *Proc. IEEE INFOCOM'12*, March 2012, pp. 2671–2675.

[9] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," in *Proc. ACM MOBICOM*, August 2000, pp. 56–67.

[10] A. Camill *et al.*, "IRIS: Integrated data gathering and interest dissemination system for wireless sensor networks," *Elsevier Ad Hoc Networks*, vol. 11, no. 2, pp. 654 – 671, 2013.

[11] G. Tolle and D. Culler, "Design of an application-cooperative management system for wireless sensor networks," in *Proc. of European Workshop on Wireless Sensor Networks'05*, 2005, pp. 121–132.

[12] P. Levis *et al.*, *Ambient Intelligence*. Springer Berlin Heidelberg, 2005, ch. TinyOS: An Operating System for Sensor Networks, pp. 115–148.

[13] S. S. Kulkarni and M. Arumugam, "Infuse: A tdma based data dissemination protocol for sensor networks," *International Journal of Distributed Sensor Networks*, vol. 2, no. 1, pp. 55 – 78, 2006.

[14] C. Shanti and A. Sahoo, "Treefp: A tdma-based reliable and energy efficient flooding protocol for wsns," in *Proc. of IEEE WoWMoM'11*, 2011, pp. 1–7.