Technical Report

Exploiting a Prioritized MAC Protocol to Efficiently Compute Min and Max in Multihop Networks

Björn Andersson

Nuno Pereira

Eduardo Tovar

HURRAY-TR-070502

Version: 0

Date: 05-25-2007

Exploiting a Prioritized MAC Protocol to Efficiently Compute Min and Max in Multihop Networks

Björn Andersson, Nuno Pereira, Eduardo Tovar

IPP-HURRAY!

Polytechnic Institute of Porto (ISEP-IPP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8340509

E-mail: {bandersson, npereira, emt}@dei.isep.ipp.pt

http://www.hurray.isep.ipp.pt

Abstract

Consider a wireless sensor network (WSN) where a broadcast from a sensor node does not reach all sensor nodes in the network; such networks are often called multihop networks. Sensor nodes take sensor readings but individual sensor readings are not very important. It is important however to compute aggregated quantities of these sensor readings. The minimum and maximum of all sensor readings at an instant are often interesting because they indicate abnormal behavior, for example if the maximum temperature is very high then it may be that a fire has broken out. We propose an algorithm for computing the min or max of sensor reading in a multihop network. This algorithm has the particularly interesting property of having a time complexity that does not depend on the number of sensor nodes; only the network diameter and the range of the value domain of sensor readings matter.

Exploiting a Prioritized MAC Protocol to Efficiently Compute Min and Max in Multihop Networks

Björn Andersson[1], Nuno Pereira[1], and Eduardo Tovar[1]

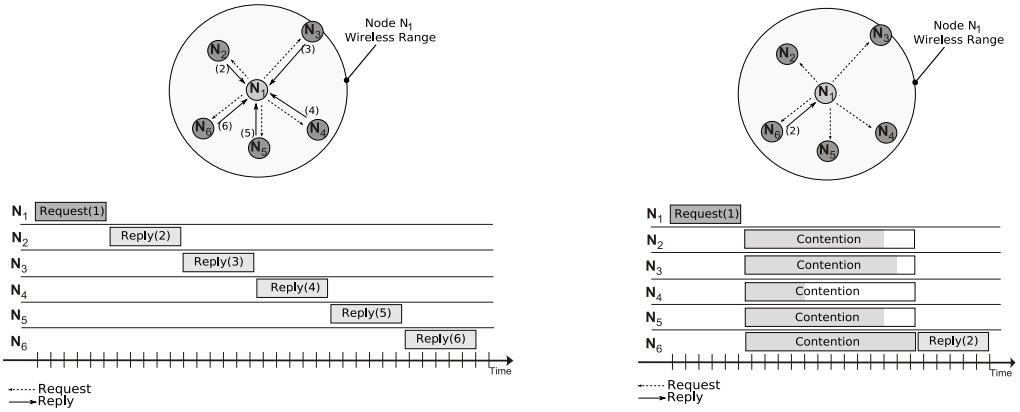
[1]IPP-Hurray! Research Group,
Polytechnic Institute of Porto,
Porto, Portugal
{bandersson,npereira,emt}@dei.isep.ipp.pt

Abstract — *Consider a wireless sensor network (WSN) where a broadcast from a sensor node does not reach all sensor nodes in the network; such networks are often called multihop networks. Sensor nodes take sensor readings but individual sensor readings are not very important. It is important however to compute aggregated quantities of these sensor readings. The minimum and maximum of all sensor readings at an instant are often interesting because they indicate abnormal behavior, for example if the maximum temperature is very high then it may be that a fire has broken out. We propose an algorithm for computing the min or max of sensor reading in a multihop network. This algorithm has the particularly interesting property of having a time complexity that does not depend on the number of sensor nodes; only the network diameter and the range of the value domain of sensor readings matter.*

1 Introduction

Wireless sensor networks (WSN) often take many sensor readings of the same type (for example, temperature readings), and instead of knowing each individual reading it is important to know aggregated quantities of these sensor readings. For example, each sensor node senses the temperature at its location, and the goal is to know the maximum temperature among all nodes at a given moment.

Several solutions for data aggregation have been proposed for multihop networks. Typically, nodes self-organize into a convergecast tree with a base station at the root [1, 2]. Leaf nodes broadcast their data. All other nodes wait until they have received a broadcast from all of their children; a node aggregates the data from its children and makes a single broadcast. Techniques have been proposed for computing useful aggregated quantities such as minimum and maximum values, the number of nodes and the median among a set of sensor nodes. They offer good performance because they exploit the opportunities for parallel transmission, and the processing enroute makes the transmitted packet typically smaller than the sum of the size of the incoming packets.



(a) Naïve Algorithm

(b) Algorithm Exploiting a Prioritized MAC

Figure 1: Computing min and max in a single broadcast domain.

Despite these optimizations, the performance is still inhibited by the fact that in a single broadcast domain, at most one packet can be sent and hence the time-complexity still depends on the number of sensor nodes. This is particularly problematic for dense networks; it has been claimed that even a small broadcast domain (covering an area $<10m^2$) may contain a few hundred sensor nodes [3]. In order to improve performance to another level, it is necessary to design distributed algorithms that circumvent this limitation.

In this paper, we propose an algorithm for computing the min or max of sensor reading in a multihop network. This algorithm has the particularly interesting property of having a time complexity that does not depend on the number of sensor nodes; only the network diameter and the range of the value domain of sensor readings matter.

We consider this result to be significant because: (i) sensor networks are designed for large scale, dense networks and it is exactly for such scenarios that our algorithms excel and (ii) the techniques that we use depend on the availability of a prioritized MAC protocol that supports a very large range of priority levels and is collision-free assuming that priorities are unique, and such a protocol has recently been proposed [4], implemented and tested [5, 6] on a sensor network platform.

The remainder of this paper is structured as follows. Section 2 gives an application background and the main idea of how a prioritized MAC protocol can be used. This section focuses on a single broadcast domain. Section 3 presents the new algorithm which offers a time-complexity that is independent of the number of sensor nodes. Section 4 discusses practical aspects of the algorithms. It also discusses the ability of previous work to solve the problem addressed in this paper. Finally, Section 5 draws conclusions.

2 Preliminaries and Motivation

The basic premise for this work is the use of a prioritized MAC protocol for wireless medium. This implies that the MAC protocol assures that of all nodes contending for the medium at a given moment, the ones with the highest priority gain access to it. As a result of the contention for the medium, all participating nodes will have knowledge of

the winner's priority. This is inspired by Dominance/Binary-Countdown protocols [7], implemented for wired networks in the widely used CAN bus [8]. In our prioritized MAC protocol for wireless medium, lower priority values mean higher priority, which is also similar to Dominance/Binary-Countdown protocols. However, such protocols assume that priorities are unique. We do not make that assumption.

The protocol in [4, 5] offers this behavior and Section 4.2 gives simply an overview of it. The focus of this paper will instead be on exploiting such a prioritized MAC protocol. We show that the availability of such a protocol enables efficient distributed computations of aggregated quantities in WSN.

2.1 Motivation and the Main Idea

The problem of computing aggregated quantities in a single broadcast domain can be solved with a naïve algorithm: every node broadcasts its sensor reading. Hence all nodes know all sensor readings and then they can compute the aggregated quantity. This has the drawback that in a broadcast domain with m nodes, at least m broadcasts are required to be performed. Considering that WSN are designed for large scale, dense networks [9, 3], the naïve approach can be inefficient; it causes a large delay and the long execution time wastes energy.

Let us consider the simple application scenario depicted in Figure 1(a), where a node (node N_1) needs to know the minimum temperature reading among its neighbors. Let us assume that no other node attempts to access the medium before this node. A naïve approach would imply that N_1 broadcasts a request to all its neighbors and then waits for the corresponding replies from them. As a simplification, assume that nodes have set up a scheme to orderly access the medium in a time division multiple access (TDMA) fashion, and that the initiator node knows the number of neighbor nodes. Then N_1 can compute a waiting timeout for replies based on this knowledge. Clearly, with this approach, the execution time depends on the number of neighbor nodes (m).

Consider now that a prioritized MAC protocol such as the one described in the beginning of Section 2 is available. This alternative would allow an approach as depicted in Figure 1(b). Assume that the range of the analog to digital converters (ADC) on the sensor nodes is known, and that the MAC protocol can, at least, represent as many priority levels. Now, to compute the minimum temperature among its neighbors, node N_1 needs to perform a broadcast request that will trigger all its neighbors to contend for the medium using the prioritized MAC protocol. If neighbors access the medium using the value of their temperature reading as the priority, the priority winning the contention for the medium will be the minimum temperature reading. (The different length of the gray bars inside the boxes depicting the contention in Figure 1(b) represent the amount of time that the node actively participated in the medium contention.) With this scheme, more than one node can win the contention for the medium. But considering that as a result of the contention, nodes will know the priority of the winner, no more information needs to be transmitted by the winning node.

In this scenario, the time to compute the minimum temperature reading only depends on the time to perform the contention for the medium, not on m.

A similar approach can be used to compute the maximum temperature reading. Instead of directly coding the priority with the temperature reading, nodes will use the bitwise

Algorithm 1 Computing MIN, this algorithm should be run at deployment

1: Run a topology discovery algorithm. Find the interference graph and the communication graph.
2: Partition the network such that all sensor nodes in a partition $PART_p$ can reach each other with a single broadcast.
3: Assign a timeslot to each partition such that if two nodes N_i and N_j belonging to different partitions are assigned to the same time slot then the range between N_i and N_j must be large enough to ensure that they do not interfere.
4: For each partition, elect a *partition leader*.
5: Among the partition leaders in step 4, elect a leader.
6: Create a convergecast tree from every partition leader to the leader.

Algorithm 2 Computing MIN, this algorithm should be run at run-time

1: Each sensor nodes N_i takes a sensor reading. Let v_i denote this sensor reading.
2: Each node N_i in $PART_j$ waits until the time slot SLOT($PART_j$) and then it sends an empty packet with the priority given by v_i. After the tournament the sensor node gets the winning priority in this slot. Let $winnerprio_i$ denote.
3: Use any convergecast algorithm (see for example [10]) to communicate the results $winnerprio_i$ from partition leaders to the leader.
4: The leader takes the min of all $winnerprio_i$ that it receives. This minimum is the minimum of all sensor readings.

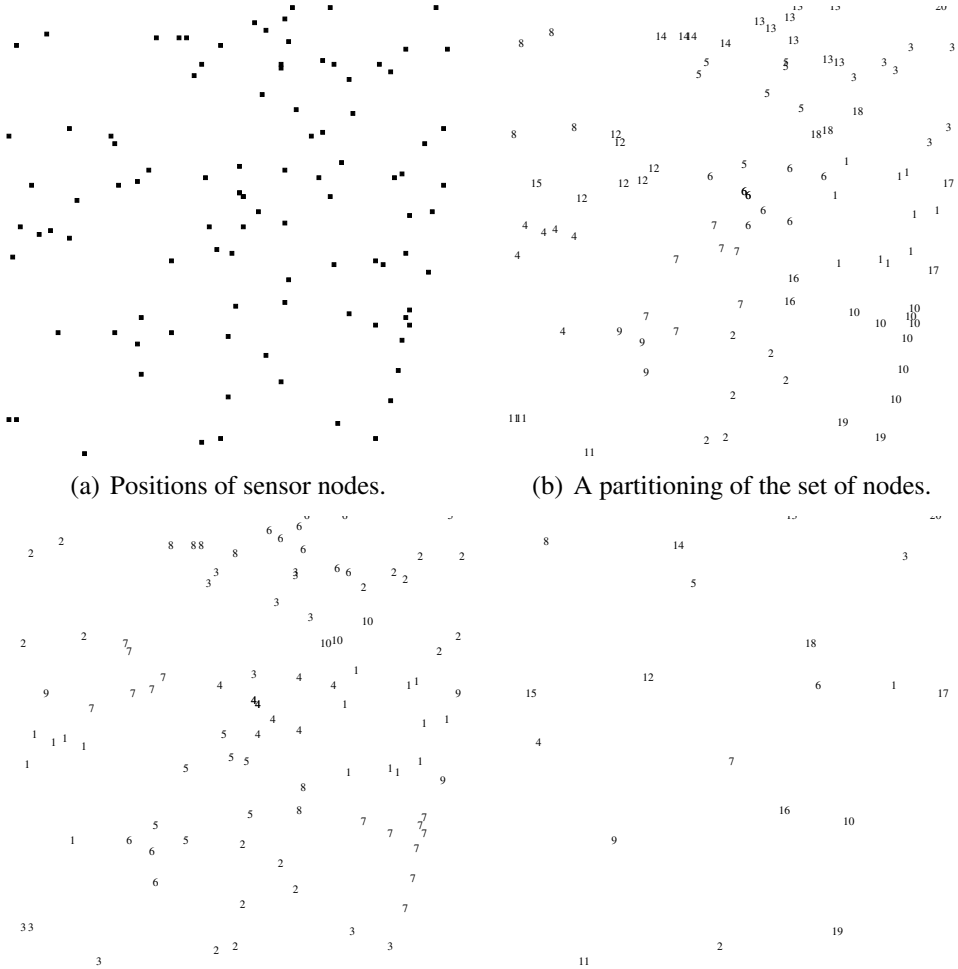
negation of the temperature reading as the priority. Upon completion of the medium access contention, given the winning priority, nodes perform bitwise negation to know the maximum temperature value.

3 The new algorithm

It should be clear that the algorithms for computing min and max in a single broadcast domain (presented in Section 2.1) does not work in a multihop network. In this section, we will extend them.

We assume that the network topology is known and that nodes do not move. We also assume that time is slotted such that all nodes know the time when a time slot begins and they also know the identifier of the time slot. One way to implement that is to use a sensor node platform that is equipped with an AM receiver that detects signals from a atomic clock. Such AM receivers are used in the FireFly mote [11] and it receives time-sync signals with a continental wide coverage. Two of them are located in Europe; one of them [12] is located in USA. It is assumed that the duration of the time slot is equal to the time it takes to run a tournament in the MAC protocol. In order to simplify the discussion, we focus on the computation of min of sensor readings; the max of sensor readings can be designed analogously to what was described in Section 2.1.

It is also assumed that all sensor nodes know when the computation should start. We think the most natural way of doing this is to do it periodically (for example, let all nodes start this computation at the beginning of a time slot such that the identifier of a time slot is divisible by 100). This is sensible for applications that continuously detect fire. But



(a) Positions of sensor nodes.

(b) A partitioning of the set of nodes.

(c) Assigning time slots to partitions.

(d) Leaders are elected.

Figure 2: A running example of the algorithm to be run at deployment.

in a multi-tiered architecture, where some nodes has a longer communication range, it is possible to let the more high-powered sensor nodes initiate a computation as well; this assumes that those high powered sensor nodes have a communication range that covers the entire network.

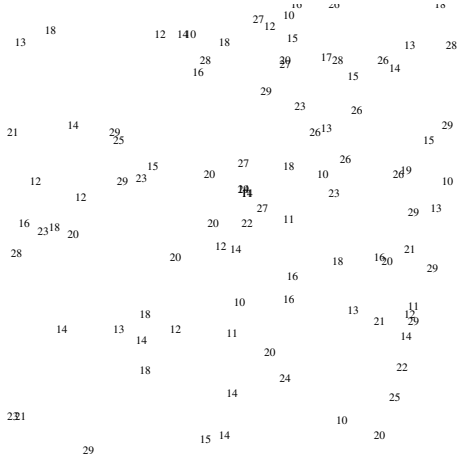
The algorithm is given as a sequence of steps shown in Algorithm 1 (which should be run at deployment) and Algorithms 2 (which should be run at run-time).

3.1 A Running Example

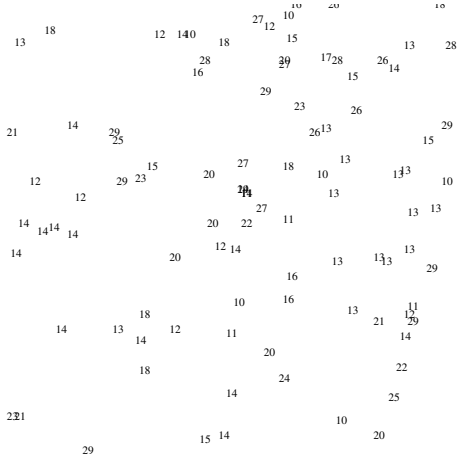
We will illustrate the algorithm with a simple example. Figure 2(a) shows a sensor network consisting of 100 nodes[1] each one depicted as a black filled square.

Let us consider the algorithm that is run when the sensor network is deployed (Algo-

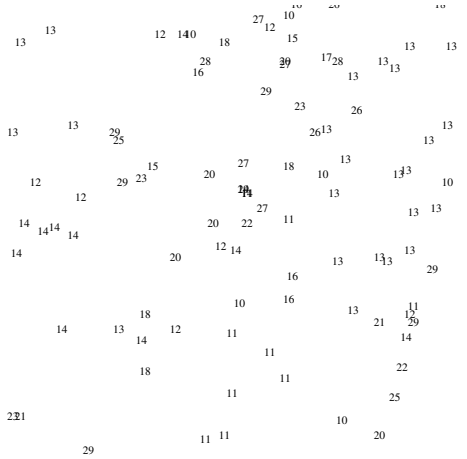
[1]Considering an example with very few nodes would not illustrate the reason why the algorithm is fast. Considering an example with many nodes would make illustration cluttered. We found that considering an example of 100 nodes is a good compromise.



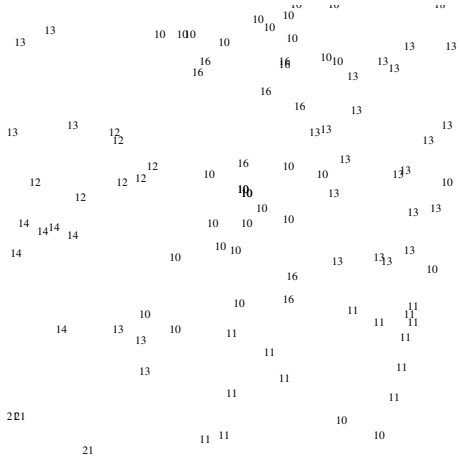
(a) Each sensor node and the temperature reading that it gets.



(b) After time slot 1.



(c) After time slot 2.

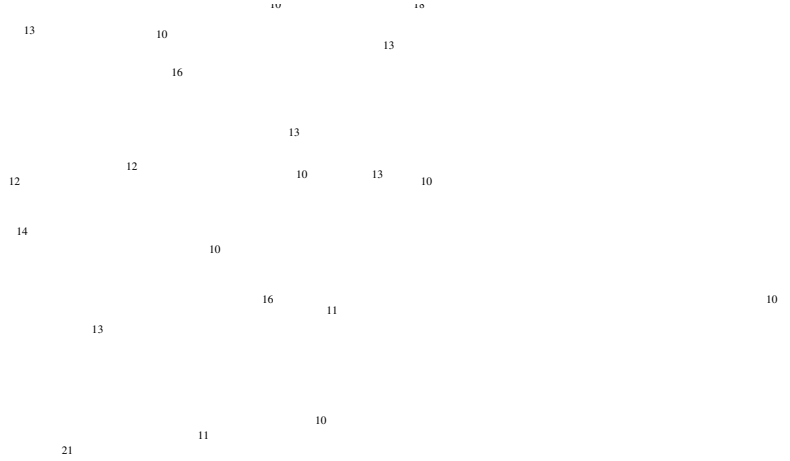


(d) After time slot 10.

Figure 3: A running example of the algorithm to be run at run-time (early steps of the algorithm).

rithm 1). First a topology discovery algorithm is ran. Then the set of nodes are partitioned. We assign a partition-id to each node; Figure 2(b) shows the sensor nodes but now a figure, its partition-id indicates its position. Then assign time slots to each partition such that if two sensor nodes, in different partitions but in the same time slot, broadcast simultaneously, then there is no collision. Figure 2(c) shows the position of each sensor node but instead of indicating the position using a rectangle, the position is indicated with the time slot id of the node. Then, we elect leaders in each partition. Figure 2(d) illustrates those leaders. Finally, we elect a leader among all nodes; we let the leader of partition 16 be that leader.

Let us consider the algorithm that is run at run-time. Figure 3(a) shows the nodes. Each node is represented by a number which is the temperature at that node. Nodes compete for the channel using their temperature readings as the priority and nodes do this in their assigned time slot. After this competition, all nodes know the minimum of temperature



(a) All partition-leaders know the minimum temperature in their partition.

(b) Partition-leaders perform convergecast to the leader and the leader knows the minimum temperature.

Figure 4: A running example of the algorithm to be run at run-time (late steps of the algorithm).

in the partition. Figure 3(b) shows the result after the first time slot. Observe that in the left-hand side of Figure 3(b) the figures "'14"' are shown. This is because partition 4 is assigned time slot 1 and in partition 4 there is a sensor node that got the temperature reading 14 and this reading is spread to all sensor nodes in partition 4 at the end of time slot 1.

Figure 3(c) shows the result after the second time slot. After 10 times slots, all nodes have broadcasted their temperature reading. Figure 3(d) shows the result after the 10:th time slot. Now, every leader of a partition knows the minimum temperature in the partition. Figure 4(a) shows that. Finally, nodes perform convergecast to the leader of the entire network. In this case, the leader of the entire network is the leader of the partition 16. After the convergecast, the leader knows that the minimum temperature in the entire network is 10. This is shown in Figure 4(b).

So far we have assumed that all transceivers can only transmit in a pre-specified channel. But many wireless standards, such as 802.11, allow a transceiver to transmit on any channel. This feature can be used advantageously by assigning each partition its own channel (instead of assigning a time slot to a partition) and this reduces the time required to perform step 2 in Algorithm 2.

4 Discussion and Previous Work

4.1 Previous work

4.1.1 MIN, MAX and General issues

A prioritized MAC protocol is useful to schedule real-time traffic [4, 5] and it can support data dissemination when topology is unknown [13]. In this paper we have discussed how to efficiently compute aggregated quantities using a prioritized MAC protocol. Distrib-

uted calculations have been performed in previous research. It has been observed [14, 15] that nodes often detect an event and then need to spread the knowledge of this event to their neighbors [14]. This is called one-to-k communication [14] because only k neighbors need to receive the message. After that, the neighbor nodes perform local computations and report back to the node that made the request for 1-to-k communication. This reporting back is called k-to-1 communication. Algorithms for both 1-to-k and k-to-1 communication are shown to be faster than a naïve algorithm but, unfortunately, the time-complexity increases as k increases. Our algorithms compute a function f and take parameters from different nodes; this is similar to the average calculations in [16]. However our algorithms are different from [14, 15, 16]; our algorithms have a time-complexity that is independent of the number of nodes.

One way to use these algorithms is to encapsulate them in a query processor for database queries. Query processors for sensor networks have been studied in previous work [1, 2] but they are different in that they do not compute aggregated quantities as efficiently as we do. They assume one single sink node and that the other nodes should report an aggregated quantity to this sink node. The sink node floods its interest in the data it wants into the network and this also causes nodes to discover the topology. When a node has new data, it broadcasts this data; other nodes hear it, then it is routed and combined so that the sink node receives the aggregated. These works exploit the broadcast characteristics of the wireless medium (like we do) but they do not make any assumption on the MAC protocol (and hence they do not take advantage of the MAC protocol). One important aspect of these protocols is to create a spanning tree. It is known that computing an optimal spanning tree for the case when only a subset of nodes can generate data is equivalent to finding a Steiner-tree, a problem known to be NP-hard (the decision problem is NP-complete, see page 208 in [17]). For this reason, approximation algorithms have been proposed [18, 19]. However, in the average case, very simple randomized algorithms perform well [20]. Since a node will forward its data to the sink using a path which is not necessarily the shortest path to the sink, these protocols cause an extra delay. Hence, there is a trade-off between delay and energy-efficiency. To make this trade-off, a framework based on feedback was developed [21] for computing aggregated quantities. Techniques to aggregate data in the network such that the user at the base station can detect whether one node gives faked data has been addressed as well [22].

Common to these previous works is that the time-complexity increases with the number of sensor nodes.

4.2 Practical issues

The MAC protocol exploited in this work was idealized based on an existing family of MAC protocols. This family is named Dominance/Binary-Countdown [7] protocols.

In the prioritized MAC protocol in [4, 5] (inspired by Dominance/Binary-Countdown protocols), nodes perform a tournament as depicted in Figure 5 to access the medium. The nodes start by agreeing on an instant when the tournament starts. Then nodes transmit the priority bits starting with the most significant bit. A bit is assigned a time interval. If a node contends with a dominant bit (“0”), then a carrier wave is transmitted in this time interval; if the node contends with a recessive bit (“1”), it transmits nothing but listens. At the beginning of the tournament, all nodes have the potential to win, but if a node

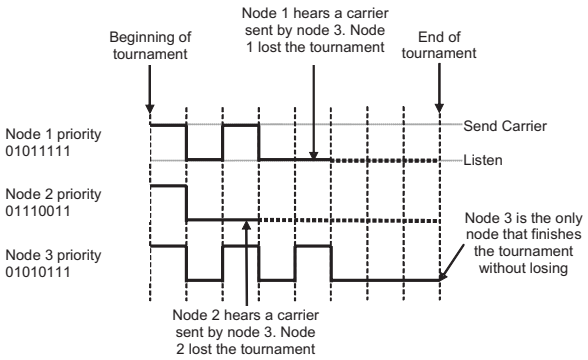


Figure 5: The MAC protocol tournament.

contends with a recessive bit and perceives a dominant bit then it withdraws from the tournament and cannot win. If a node has lost the tournament then it continues to listen in order to know the priority of the winner. When a node finishes sending all priority bits without hearing a dominant bit when it transmitted a recessive bit, then it has won the tournament and clearly knows the priority of the winner. Hence, lower numbers represent higher priorities.

We have assumed that all nodes start the execution of the protocol simultaneously. This can be dealt easily by letting a node broadcast a message containing a request to compute the aggregated quantity. All nodes receive this at approximately the same time. There are small differences in time when nodes start the protocol, but the MAC protocol (see [5]) synchronizes so that the tournament on all nodes executes simultaneously, so this poses no problem.

To support the hypothesis of implementing a protocol with similar properties for wireless networks, we have referenced the reader to [5]. So far, the implementation of this prioritized MAC protocol for wireless networks introduces a significant amount of overhead. This overhead is to a large extent due to the transition time between transmission and reception. The platform used to implement the MAC protocol in [5] had a switching time of 192μs. But this is a technological parameter that can be improved with better radio hardware, as witnessed by the fact that the Hiperlan standard [23] required a switching time of 2μs.

5 Conclusions

We have shown how to use a prioritized MAC protocol to compute aggregated quantities efficiently. The algorithms designed to exploit such MAC protocol have a time-complexity that is independent of the number of sensor nodes. This is clearly important for WSN applications that operate under real-time constraints. But, since the high speed makes it possible for nodes to stay awake for only a short time and they can then sleep, it is also very useful for reducing energy-consumption; and this gives nodes a longer life-time.

We left three important questions open (i) Is it possible to achieve the same efficiency without partitioning the network into broadcast domains? (ii) Can a similar technique be used to compute more complex aggregated quantities (such as COUNT, MEDIAN

and interpolation) of sensor readings in multihop networks? and (iii) Is the technique sufficiently reliable for large-scale systems?

Acknowledgements

This work was partially funded by the Portuguese Science and Technology Foundation (Fundação para Ciência e Tecnologia - FCT) and the ARTIST2 Network of Excellence on Embedded Systems Design.

References

[1] Y. Yao and J. Gehrke. Query processing in sensor networks. In *Proceedings of the 1st Biennial Conference on Innovative Data Systems Research (CIDR'03)*, 2003.

[2] S. Madden, M. J. Franklin, J.M. Hellerstein, and W. Hong. TAG: a tiny aggregation service for ad-hoc sensor networks. In *Proceedings of the 5th symposium on Operating systems design and implementation (OSDI'02)*, 2002.

[3] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-Efficient Communication Protocol for Wireless Microsensor Networks. In *Proceedings of the 33rd Hawaii International Conference on System Sciences (HICSS'00)*, volume 8, pages 3005–3014, Maui, U.S.A., 2000.

[4] B. Andersson and E. Tovar. Static-priority scheduling of sporadic messages on a wireless channel. In *Proceedings of the 9th International Conference on Principles of Distributed Systems (OPODIS'05)*, Pisa, Italy, 2005.

[5] N. Pereira, B. Andersson, and E. Tovar. Implementation of a dominance protocol for wireless medium access. In *Proceedings of the 12th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'06)*, Sydney, Australia, 2006.

[6] http://www.hurray.isep.ipp.pt/activities/widom/widom.zip.

[7] A. K. Mok and S. Ward. Distributed broadcast channel access. *Computer Networks*, 3:327–335, 1979.

[8] Bosch. *CAN Specification, ver. 2.0, Bosch GmbH, Stuttgart*, 1991.

[9] A. Arora. Exscal: Elements of an extreme scale wireless sensor network. In *Proceedings of the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'05)*, pages 102–108, Washington, DC, USA, 2005. IEEE Computer Society.

[10] S. Gandham, Y. Zhang, and Q. Huang. Distributed minimal time convergecast scheduling in wireless sensor networks. In *26th IEEE International Conference on Distributed Computing Systems (ICDCS'06)*, 2006.

[11] R. Mangharam, A. Rowe, and R. Rajkumar. Firefly: A cross-layer platform for wireless sensor networks. *Real Time Systems Journal, Special Issue on Real-Time Wireless Sensor Networks*, 2006.

[12] http://tf.nist.gov/stations/wwvb.htm.

[13] B. Andersson, N. Pereira, and E. Tovar. Disseminating data using broadcast when topology is unknown. In *26th IEEE Real-Time Systems Symposium (RTSS'05), Work-in-Progress Session*, pages 61–64, 2005.

[14] R. Zheng, L. Sha, and W. Feng. MAC layer support for group communication in wireless sensor networks. In *Proceedings of the second Mobile Adhoc and Sensor Systems Conference*, page 8. IEEE, 2005.

[15] K. Jamieson, H. Balakrishnan, and Y. C. Tay. Sift: a MAC protocol for event-driven wireless sensor networks. In *Proceedings of the third European Workshop on Wireless Sensor Networks (EWSN'06)*, pages 260–275. IEEE, 2006.

[16] D.S. Scherber and H.C. Papadopoulos. Distributed computation of averages over ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 23(4):776– 787, 2005.

[17] M. R. Garey and D. S. Johnson. *Computers and Intractability A guide to the Theory of NP-Completeness*.

[18] B. Krishnamachari, D. Estrin, and S. B. Wicker. The impact of data aggregation in wireless sensor networks. In *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS'02)*, pages 575 – 578. IEEE, 2002.

[19] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann. Impact of network density on data aggregation in wireless sensor networks. In *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS'02)*, page 457, Washington, DC, USA, 2002. IEEE Computer Society.

[20] M. Enachescu, A. Goel, R. Govindan, and R. Motwani. Scale-free aggregation in sensor networks. *Theoretical Computer Science*, 344(1):15–29, 2005.

[21] T. Abdelzaher, T. He, and J. Stankovic. Feedback control of data aggregation in sensor networks. In *Proceedings of the 43rd IEEE Conference on Decision and Control (CDC'04)*, pages 1490–1495 Vol.2. IEEE Computer Society, 2004.

[22] B. Przydatek, D. Song, and A. Perrig. SIA: Secure information aggregation in sensor networks. In *Proceedings of the 1st ACM International Conference on Embedded Networked Sensor Systems (SenSys'03)*, pages 255–265, 2003.

[23] ETSI (European Telecommunications Standards Institute). *Broadband Radio Access Net.(BRAN); HIPERACCESS; PHY protocol specification*.