Technical Report

Delay-Bounded Medium Access for Unidirectional Wireless Links

Björn Andersson

Nuno Pereira

Eduardo Tovar

TR-070103

Version: 1.0

Date: January 2007

Delay-Bounded Medium Access for Unidirectional Wireless Links

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 network where links may be unidirectional, that is, a computer node A can broadcast a message and computer node B will receive this message but if B broadcasts then A will not receive it. Assume that messages have deadlines. We propose a medium access control (MAC) protocol which replicates a message in time with carefully selected pauses between replicas, and in this way it guarantees that for every message at least one replica of that message is transmitted without collision. The protocol ensures this with no knowledge of the network topology and it requires neither synchronized clocks nor carrier sensing capabilities. We believe this result is significant because it is the only MAC protocol that offers an upper bound on the message queuing delay for unidirectional links without relying on synchronized clocks.

Delay-Bounded Medium Access for Unidirectional Wireless Links[1]

Björn Andersson, Nuno Pereira, Eduardo Tovar
IPP Hurray Research Group
Polytechnic Institute of Porto, Portugal
{bandersson, npereira, emt}@dei.isep.ipp.pt

Abstract

Consider a wireless network where links may be unidirectional, that is, a computer node A can broadcast a message and computer node B will receive this message but if B broadcasts then A will not receive it. Assume that messages have deadlines. We propose a medium access control (MAC) protocol which replicates a message in time with carefully selected pauses between replicas, and in this way it guarantees that for every message at least one replica of that message is transmitted without collision. The protocol ensures this with no knowledge of the network topology and it requires neither synchronized clocks nor carrier sensing capabilities. We believe this result is significant because it is the only MAC protocol that offers an upper bound on the message queuing delay for unidirectional links without relying on synchronized clocks.

1 Introduction

Consider a computer node A that can broadcast a message and computer node B that can receive this message but if B broadcasts then A cannot receive it. We say that the network topology has a *unidirectional link* from A to B. Empirical data show that unidirectional links exist and they are not uncommon; typically, in networks with low-power radios, 5-15% of all links are unidirectional [1-7]. This has been recognized at the routing layer but the MAC layer is still poorly developed for unidirectional links. Traditional MAC protocols fail on unidirectional links. For example, consider a node A that performs carrier-sensing before it sends a message to node B. Node B transmits as well but due to the fact that the link A→B is unidirectional, node A perceives that there is no carrier. Consequently, node A transmits and it collides with the transmission from node B. Also, RTS/CTS (Request-to-Send/Clear-to-Send) exchanges fail as well because node A sending a RTS-packet does not receive a CTS packet from node B. In addition, protocols that allow collisions but let a sender A wait for an acknowledgement from node B can fail too. Node B received the message but since the link A→B was unidirectional, node B cannot send an acknowledgement back to the sender A: the sender A has to wait forever. The only existing solutions today for medium access in the presence of unidirectional links require synchronized clocks [8] or cause unbounded number of collisions.

In this paper we study medium access of wireless links which may be unidirectional. We show, informally, that designing a collision-free MAC protocol is impossible. For this reason, we design a replication scheme; every message that an application requests to transmit is replicated in time by the MAC protocol with carefully selected pauses between the transmissions of replicas. This guarantees that for every message, *at least one of its replicas is transmitted without collision*.

The protocol proposed in this paper is quite heavy-weight. We will see that such a high overhead is necessary in order to bound the number of collisions and hence to achieve real-time guarantees in the presence of unidirectional links; this is our main focus. But less time-critical applications (such as file transfer) demand high throughput and networks often have links that are mostly bidirectional, and unidirectional only occasionally. In such networks, the robustness and delay guarantees offered by the bounded number of collisions of our scheme is not worth the high overhead. For this reason, we will discuss how the protocol can be adapted to obtain an *average-case* overhead similar to "normal" protocols designed for bidirectional links, while retaining the upper bound on the number of collisions in the presence of unidirectional links.

The remainder of this paper is organized as follows. Section 2 presents the system model, the impossibility of designing a collision-free MAC protocol and the main idea of the protocol. Section 3 presents schedulability analysis of sporadic message streams. Section 4 presents implementation and experimental validation of the protocol. Section 5 reviews previous work and discusses unidirectional links in its larger context. Finally, Section 6 offers conclusions.

[1] Due to space limitations, some results in this conference version are omitted. See the extended version for details [11].

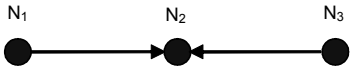


Fig. 1. A network topology which illustrates the impossibility of collision-free medium access in the presence of unidirectional links. N_1 can transmit to N_2 but N_2 cannot transmit to N_1. Analogously for N_2 and N_3. When N_1 and N_3 transmit there will be a collision on node N_2.

2 Preliminaries and the Main idea

2.1 Network and Message Model

The network topology is described using a graph with nodes and links. A node represents a computer node. A link is directed. Consider a node N_i that broadcasts a message or any signal (for example an unmodulated carrier wave). Then node N_k will receive it if and only if there is a link in the topology graph from node N_i to node N_k. A node can only transmit by performing a broadcast and it is impossible for a node N_i to broadcast such that only a proper subset of its neighbor nodes receive it. No assumption on the topology of each node is made. It is allowed that a node has only outgoing links or only ingoing links or no links at all. Unless otherwise stated, the topology is assumed to be unknown to the MAC protocol. In Section 5, we will discuss how knowledge of the network topology can be exploited.

Let *mtotal* denote the number of nodes and let *m* denote the number of nodes that can transmit. Nodes are indexed from 1 to *mtotal*, where the *m* nodes that can transmit have the lowest index. As an illustration, consider a network with $mtotal = 5$ nodes but 2 nodes will never transmit; these nodes will have index 4 and 5. The other $m = 3$ nodes are permitted to request to transmit and these nodes have index 1, 2 and 3.

We will initially assume that on each node with index $1..m$, there is a single application and it makes only a single request to the MAC protocol to transmit a message. The exact time of the request is unknown before run-time and the MAC protocol does not know about the time of the request before it occurs. Let J_i denote this single message on node N_j. (J_i is analogous to a job in processor scheduling.) It is assumed that when the MAC protocol sends a message it takes one time unit. We are interested in finding a value z such that it holds for any node that the time from when a message transmission request is made at a node until this message is successfully transmitted without collision is at most z.

Let $prop_{i,j}$ denote the propagation delay of the medium between nodes N_i and N_k. We assume that $prop_{i,k}$ is unknown but it is bounded such that $\forall i,k \in \{1..mtotal\}: 0 < prop_{i,k} \leq prop$. Hence, *prop* is an upper bound on the propagation delay of the medium; we expect that a typical value is $prop = 1\mu s$ for distributed real-time systems in a small geographical area, such as a ship, a factory or a virtual caravan of cars. We assume that *prop* is finite but we make no assumptions on its actual value. However, we assume the following: (i) nodes can "boot" at different times and when they boot, they do not have synchronized clocks; (ii) when a node is transmitting it cannot receive anything; and (iii) the MAC protocol can be represented as a set of timed automata, with potentially different automata on different computer nodes.

2.2 Impossibility

Let us now show that, under these assumptions, it is impossible to design a collision-free MAC protocol when there are unidirectional links. Consider Figure 1. It illustrates a simple exemplifying topology. For such topology and links characteristics, it is necessary that N_1 does not transmit simultaneously with N_3, in order to guarantee that collisions will not occur. This requires that N_1 can get some information about the other nodes on whether there is an ongoing transmission on the other link. But N_1 cannot hear anything so the transmission from N_1 may overlap with the transmission from N_3, and then N_2 will not receive any of them. Hence, it is impossible to design a MAC protocol that is guaranteed to be collision-free in the presence of unidirectional links. Even if a node knows the topology but it does not know the time when other nodes will transmit then a collision can occur, and hence the above mentioned impossibility also extends to the case where the topology is known to the MAC protocol.

Given the impossibility of collision-free medium access in the presence of unidirectional links we will now design a solution.

2.3 The Main Idea

For each message J_i the MAC protocol transmits the message several times. Each one of them is called a *replica*. Of those replicas from message J_i let $J_{i,1}$ denote the one that is transmitted first. Analogously, let $J_{i,2}$ denote the one that is transmitted second, and so on. The number of replicas transmitted for each message of J_i is $nreplicas(J_i)$, and the time between the start of transmission of $J_{i,j}$ until the start of transmission of $J_{i,j+1}$ is denoted as $\Delta_{i,j}$. Figure 2 illustrates these concepts for the case when all messages request to transmit

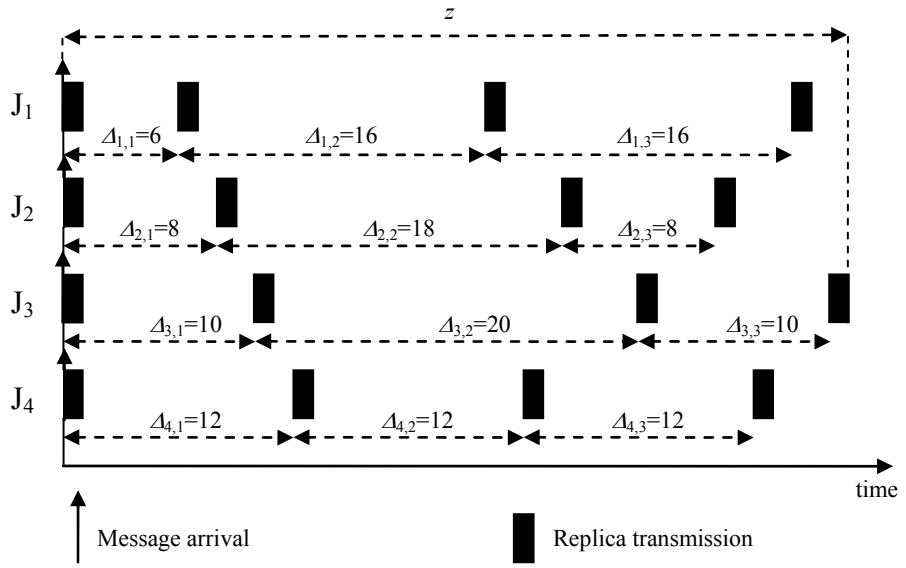


Fig. 2. Transmission of replicas with a possible assignment of Δ:s to messages. J_1, J_2, J_3, J_4 requested to transmit simultaneously at time 0. As it can be seen, at least one replica is collision-free. It turns out that for every possible combination of times of requests of J_1, J_2, J_3, J_4 this is true as well.

simultaneously. We let $J_{i,1}$ be transmitted immediately when J_i is requested to be transmitted. For convenience, we assume in this section (Section 2) that $prop = 0$ and this is known to the MAC protocol. In Section 5, we will discuss a simple technique to extend the results to the case where $prop > 0$.

We will now reason about how to select $nreplicas(J_i)$ and then select $\Delta_{i,j}$. It is necessary to select $nreplicas(J_i) \geq m$ because otherwise there is a topology for which it is possible that all replicas of J_i collide. To see this, consider m nodes where one central node N_k has ingoing links from all other nodes; one of these other nodes is node N_i. There is also a link from N_k to N_i. Let us now consider the case where N_i broadcasts its replicas. Let N_l denote any other node than N_k and N_i. The first message transmission of J_l can happen at any time, so it can collide with one of the replicas from J_i. Analogously, the first replica of another message J_l can collide with another replica of J_i. In addition, the first replica from J_k can occur any time too, so this first replica can be transmitted when J_i sends a replica to N_k. Then N_k will not hear the replica from J_i Hence, if J_i transmits $nreplicas(J_i) < m$ replicas, it can happen that none of them are received at node N_k. Therefore, $nreplicas(J_i)$ must be selected such that:

$$nreplicas(J_i) \geq m$$

Later in this section, we will select $\Delta_{i,j}$ such that at most one replica from J_i can collide with a replica of J_l. With such an assignment of $\Delta_{i,j}$, the assignment of $nreplicas(J_i)$ is as follows:

$$\forall i \in \{1,..,m\}: nreplicas(J_i) = m \quad (1)$$

Having selected nreplicas(Ji) = m, the issue of selecting □i,j will now be considered. Clearly, since a node i transmits nreplicas(Ji) replicas, it is necessary to specify nreplicas(Ji) – 1 values of □i,j for node i. Consider the time span starting from when an application requests to transmit on a node until the last replica has finished its transmission on that node. The maximum duration of this time span over all nodes is z (as mentioned in Section 2.1). Figure 2 illustrates this. Clearly, we wish to minimize z. This can be formulated as a mixed linear/quadratic optimization problem. Therefore, the objective is to minimize z subject to:

$$\forall i \in \{1,..,m\}: \sum_{j=1}^{nreplicas(J_i)-1} \Delta_{i,j} + 1 \leq z$$
$$\forall i \in \{1,..,m\}, \forall j \in \{1,..,nreplicas(\tau_i)-1\}: 0 \leq \Delta_i \quad (2)$$

and (1), and subject to an additional third constraint that will be described now. Let u and v denote the indices of two nodes that may transmit. Hence, u and v belong to the set $\{1..m\}$. Let j_u and j_v denote the indices of the first replica of the sequence of replicas transmitted in nodes N_u and N_v, respectively. Hence j_u belongs to $\{1..nreplicas(J_u)–1\}$ and j_v belongs to $\{1..nreplicas(J_v)–1\}$. Let l_u and l_v denote the lengths of these subsequences in terms of the number of replicas. l_u should be selected such that $j_u + (l_u - 1) \leq nreplicas(J_u) – 1$. Analogous for l_v. Hence, l_u belongs to $\{1.. nreplicas(J_u) – j_u\}$ and l_v belongs to $\{1.. nreplicas(J_v) – j_v\}$. We say that a combination of u, v, j_u, j_v, l_u, l_v is valid if: (i) these 6

variables are within their ranges; and (ii) $u \neq v \wedge (j_u \neq j_v \vee l_u \neq l_v)$. For every valid combination of u, v, j_u, j_v, l_u, l_v, the optimization problem must respect the following constraint:

$$\left(\left(\sum_{j=j_u}^{j_u+l_u-1}\Delta_{u,j}\right)-\left(\sum_{j=j_v}^{j_v+l_v-1}\Delta_{v,j}\right)\right)^2 \geq 2^2 \qquad (3)$$

Intuitively, (3) states that there is no sum of consecutive Δ:*s* on node u which is equal to a consecutive sum of Δ:*s* on node v. In addition, the difference is larger than 2; this implies that it is enough to be sure that there is no collision. (To understand why the difference must be 2, consider the following system: $m = 2$, $nreplicas(J_1) = 2$ and $nreplicas(J_2) = 2$ and $\Delta_{1,1} = 2$ and $\Delta_{2,1} = 3.98$, and J_1 arrives at time 0.99 and J_2 arrives at time 0. Then the first replica of J_1 and J_2 will collide, and the second replicas of J_1 and J_2 will collide as well. One can see that the sum of Δ:*s* must differ by the duration of two.).

Therefore, (3) states that at most one replica from node u can collide with a replica from node v. Hence, of those $nreplicas(J_u)$ replicas sent from node u, at most $m - 1$ of them can collide. Naturally, this permits stating Theorem 1 below.

Theorem 1. If the differences between transmission start times of replicas are selected according to (1)-(3), then it holds that: (i) for every node i, at least one replica does not collide; and (ii) the time from when an application requests to transmit on node i until the last replica is transmitted on node i is at most z.

Proof: Follows from the discussion above. □

We will now illustrate the use of (1)-(3) in Example 1.

Example 1. Consider $m = 4$ to be solved using (1)-(3). The solution that is obtained is as follows:

$$\begin{array}{lll} \Delta_{1,1}=6 & \Delta_{1,2}=16 & \Delta_{1,3}=16 \\ \Delta_{2,1}=8 & \Delta_{2,2}=18 & \Delta_{2,3}=8 \\ \Delta_{3,1}=10 & \Delta_{3,2}=20 & \Delta_{3,3}=10 \\ \Delta_{4,1}=12 & \Delta_{4,2}=12 & \Delta_{4,3}=12 \end{array}$$

This is illustrated in Figure 2. □

It is easily perceived that the number of inequalities in (3) grows as $O(m^6)$. Hence, it is only possible to solve small problems with this approach. (There were 232 constraints for $m = 4$ and 3411 constraints for $m = 6$. We used a modeling tool (AMPL [9]) and a back-end solver (LOQO [10]), and with these tools it was only possible to solve (1)-(3) for $m \leq 6$.) Many interesting systems are larger though. For those systems the optimization problem phrased in (1)-(3) simply cannot be solved because the number of inequalities in (3) is too large. (The extended version of this paper [11] presents techniques that find Δ:*s* for $m \leq 100$ by trading off the optimality of z.)

3 Sporadic Message Streams

Let us now consider that traffic is characterized by the *sporadic model* [12]. Each node has exactly one message stream. Node N_i is assigned the message stream τ_i. This message stream makes an infinite sequence of requests, and for each request, the message stream requests to transmit a message. The exact time of a request is unknown before run-time and the MAC protocol only knows about the time of the request when it occurs. But for every message stream τ_i there is at least T_i time units between two consecutive requests in τ_i and the MAC protocol knows all T_i. For every such request, the MAC protocol must finish the transmission of one replica of a message from stream τ_i without collisions at most T_i time units after the request. If this is the case, then we say that deadlines are met; otherwise a deadline is missed. Naturally, we assume $0 \leq T_i$.

From Section 2.3 it results that the maximum time it takes from when a message requests to send until the MAC protocol has transmitted a collision-free replica is z, if a message stream only makes a single request. Based on this, it would be tempting to think that if $\forall i \in \{1..m\}$: $z \leq T_i$ then all deadlines are met. Unfortunately, this is false, as illustrated by Figure 3, even if $T_1 = T_2 = T_2 = \ldots = T_m$. A correct schedulability analysis is given now.

Let w_i be defined as:

$$w_i = \sum_{j=1}^{nreplicas(\tau_i)-1}\Delta_{i,j} + 1 \qquad (4)$$

Theorem 2. If (Δ:*s* satisfy (1)-(3))$\wedge$(w_i is computed according to (4))$\wedge$($w_i \leq T_i$)$\wedge$($\forall k,k\neq i: w_i \leq T_k - w_k - 1$) then every message released from τ_i transmits at least one replica collision-free at most T_i time units after the message transmission request occurred.

Proof: Follows from the fact that during a time interval of duration $T_k - w_k - 1$, message stream τ_k can release at most one message. □

4 Implementation and Experiments

Having seen that the replication scheme can guarantee that at least one replica is collision-free in theory, we now turn to practice. We want to test the following hypotheses:

1. The replication scheme is easy to implement.
2. The number of lost or corrupted messages at the receiver is smaller when the replication scheme in this paper is used, as compared to a replication scheme with random pauses. This applies even if the random scheme transmits only a single replica per message.

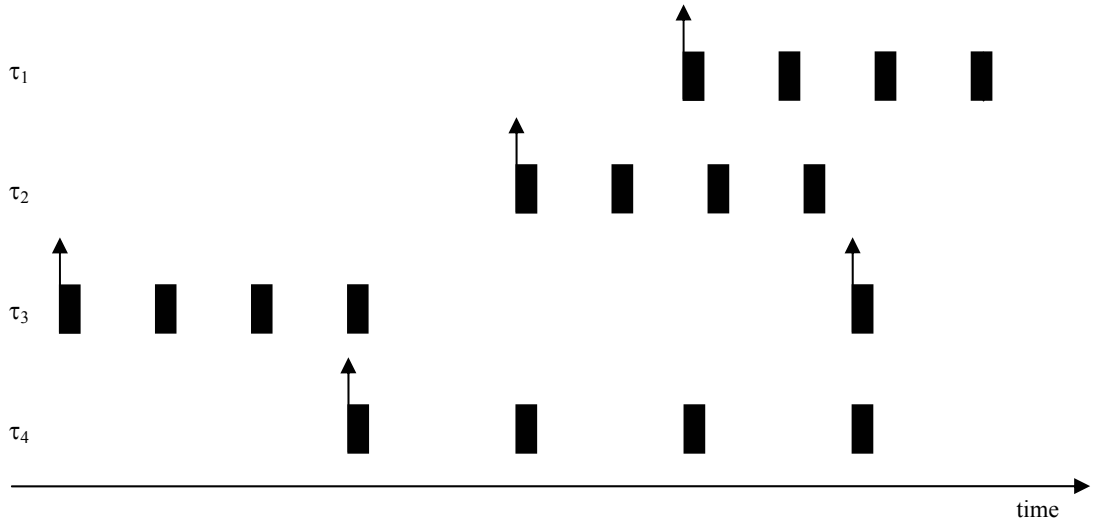


Fig. 3. Consider Δ:*s* that are selected based on the assumption a transmission request on a node occurs at most once. If these Δ:*s* are used for sporadic message streams with $T_1 = T_2 = T_3 = T_4 = z$ then a deadline miss can occur. All replicas from τ_4 collide and τ_4 misses its deadline.

3. The replication scheme guarantees that for each message, at least one replica is indeed collision-free.
4. If a link is bidirectional then our replication scheme can be extended so that it still offers a bounded number of collisions but it also has a low average-case overhead.

In order to test these hypotheses, we implement the replication protocol both on a real platform and use simulation (for details, see [11]). The following sections describe the implementation, experimental setup and results obtained. For these experiments, we used more than 6 computer nodes and hence the optimal algorithm described in Section 2.3. could not be used. We developed a heuristic algorithm (see [11]) for assigning Δ:*s* such that (1)-(3) are satisfied.

4.1 Implementation and Experimental Setup

The replication protocol was implemented on the MicaZ platform [13] and this implementation was dubbed HYDRA. MicaZ is a platform offering a low power microcontroller, 128 Kbytes of program flash memory and an IEEE 802.15.4 compliant radio transceiver, capable of 250 kbps data rate. The MicaZ supports running TinyOS [14] an open-source operating system. This platform was found to be attractive for the implementation of our experiments because of some particularly relevant characteristics: (i) it allowed us to replace the MAC protocol; (ii) the timers available where reasonably precise for our application; (iii) the radio transceiver makes automatic CRC checks and inserts a flag indicating the result of this check along with the packet, and (iv) the spread spectrum modulation used makes data frames resistant to noise and distortion. Hence, collisions due to medium access are the main source of lost frames or corrupted frames.

The experimental application setup consisted of one receiving node and a many sending nodes. Efforts where made such that the experiments took place under a similar, noise-free, environment. The sending nodes send messages with sequence numbers so that the receiving node detects when a message has been lost. Additionally, the receiver collected other statistics, such as total number of replicas and redundant replicas received (by redundant replicas we mean replicas for which a previous replica of the same message has already been received). The time to transmit a replica is 928μs. So, we let one time unit represent 1 ms to improve robustness against propagation delay and clock inaccuracy.

First, to acquire the probability that a replica is not correctly received (this is due to noise or distortion), we set up a scenario with one sending node (N1) and one receiving node (N2). Node N1 transmitted 2 replicas per message and N2 gathered statistics on the number of received replicas. We obtained that the probability of having a replica lost is approximately 0.002737%. If the events "a replica is lost" were independent, we would expect that the probability that two consecutive replicas are lost is 0.000027372. Hence we would expect the probability that a message was lost is 0.000027372 as well. However, we observed a 0.00153% probability for messages loss; this indicates that errors are correlated, which was expected.

After that, we ran experiments with different number of nodes, for three different MAC protocols: (i) one where we use our scheme with deterministic Δ:*s* (HYDRA); (ii) another where we used a similar scheme, but where the Δ:*s* were random variables within an

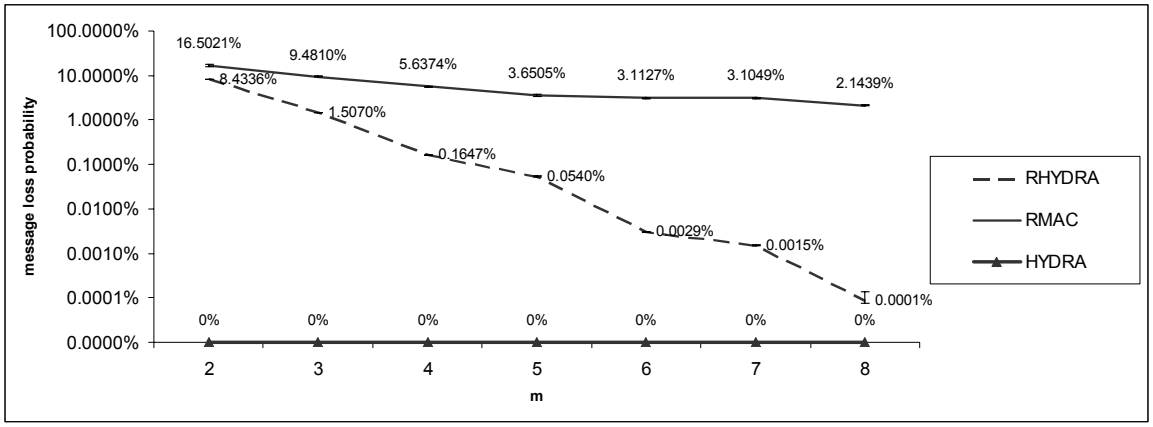


Fig. 4. Message loss ratio in simulation

interval between 1 ms and $(T_i - 1)/(nreplicas(\tau_i) - 1)$ time units, which was named Random HYDRA (RHYDRA) and (iii) finally a third MAC protocol where only one replica is sent at a random time within the interval $[0, T_i - 1]$ time units after the message was requested, which will be referred to as Random MAC (RMAC). The Δ:*s* were obtained from a close-to-optimal algorithm (see [11] for details). From these Δ:*s*, we derived z and T_i. The application on N_i generated message transmission requests such that the time between two consequtive requests is a uniform random variable with minimum T_i and maximum $T_i \times 1.25$.

The experiments where performed until each node transmitted 100000 messages, for $m = 2$ and $m = 4$. The resulting message loss rate is shown in Figure 5, which is presented in a logarithmic scale. By these results, we can observe that HYDRA obtained a message loss rate always better to the replica loss rate (0.002737%) previously obtained, indicating that noise was the cause for application message loss.

Performing statistically significant experiments with the actual implementations was very time consuming. Therefore, in order to test our protocol further, a simulation model for the protocol in OMNeT++ [15] was implemented. With this model we study the message loss ratio for different numbers of nodes with HYDRA, RHYDRA and RMAC (see [11] for details). The simulator assumes that replicas cannot get lost or corrupted due to noise, but it does model collisions which is the only source of lost messages.

All simulations were executed for a length of 10 simulated hours. For simulations involving random numbers generation, several independent runs were executed to verify the statistical validity of the results. The results of the simulations are given in Figure 4 with respective error bars which are mostly not visible due to the small variation found throughout the simulation runs.

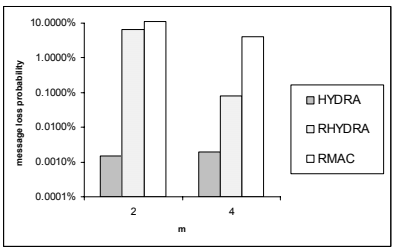


Fig. 5. Message loss ratio of experiments with MicaZ platforms

Observe that the application message loss for the scheme using deterministic Δ:s is always zero. This is expected as the simulation only models collisions, no noise in transmission was introduced, whereas the other schemes suffer from application message loss.

4.2 Support of Hypotheses

§Hypothesis 1. In order to test Hypothesis 1 the time required to implement HYDRA was measured. We spent approximately 3 days on implementing the protocol. Almost a third of this time was spent on getting familiar with the platform details. The time for coding the protocol was less than a day and we encountered no relevant bugs that were related to the implementation of the protocol. However, we encountered and fixed some bugs related to the platform. This suggests that Hypothesis 1 withstood our test.

§Hypothesis 2. The experiments presented in Section 4.1. corroborate Hypothesis 2.

§Hypothesis 3. Testing Hypothesis 3 is difficult because it is difficult to know if a lost frame is due to a collision or due to noise/distortion. Corrupt CRC may be because of noise or it may be because of collisions. Based on the experiments with the actual implementation of HYDRA in Section 4.1, it results that the number of lost messages is less than the probability

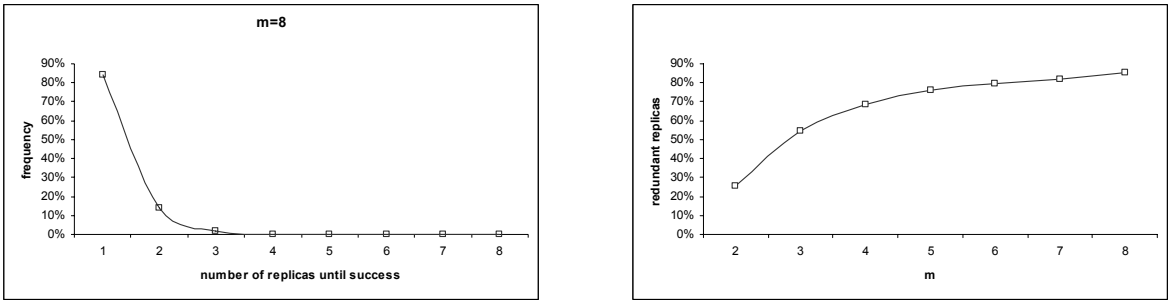


Fig. 6. The frequency of the number of necessary replicas and variation of the number of redundant replicas with m.

of a single message with a single sender being lost; this corroborates our hypothesis that the implementation of our protocol indeed guarantees that at least one replica is collision-free. Furthermore, we have run simulations during a period of 100 simulated hours for the scheme using deterministic Δ:s for $2 \le m \le 8$ and found that no application messages were lost during these simulation runs. This suggests that Hypothesis 3 withstood our test.

§Hypothesis 4. In order to test Hypothesis 4, we considered the simulation experiments used to test Hypothesis 3 and acquired both the frequency of the number of replicas necessary until the first replica is transmitted without collision (Figure 6a) and the number of redundant replicas for $2 \le m \le 8$ (Figure 6b). Observe, in Figure 6b, that for the case with $m = 8$ we obtain that approximately 84% of the first replicas of a message are collision-free. Hence, if most (but not all) links are bidirectional and we would have used a scheme where the receiver sends an acknowledgement when it receives the first successful replica then in approximately 84% of the cases the sender N_s only needs to send one replica. Hence, in 84% of the cases, N_s can send 7 non real-time messages instead of the replicas that N_s would normally send. This discussion supports, Hypothesis 4.

In order for the acknowledgement scheme described above to be efficient, it is necessary that the time required to send acknowledgements is negligible. Nonetheless, it could easily been the case by using longer packets (say 1500 bytes) for data and short packets (say 20 bytes) for the acknowledgements. But unfortunately this is not supported by our experimental platform so we did not implement it.

5 Discussion and Previous Work

Bidirectional links are useful for MAC and routing protocols. Let us categorize a MAC protocol based on whether it can suffer from collisions. If it can suffer from collisions then a sender typically retransmits data packets until it receives an acknowledgement from the intended receiver. Typically the data and the acknowledgement are transmitted on the same link, so this requires bidirectional links. This is exemplified by ALOHA [16] and some CSMA/CA protocols. MAC protocols that are collision-free typically rely on that senders receive feedback from the intended receiver. Some protocols, such as MACA [17] do this using an RTS/CTS exchange before the data packet is sent. In other protocols, a receiver sends a busy tone when it receives a packet and other senders can hear it, thus avoiding a collision. Common to all these MAC protocols is that they depend on bidirectional links. Routing algorithms also typically assume that links are bidirectional, being one notable exception the Dynamic Source Routing (DSR) [18]. We can conclude that the current communication protocols are heavily dependent on bidirectional links.

Unfortunately, unidirectional links are not rare and they are caused by a variety of reasons such as: (i) differences in antenna and transceivers even from the same type of devices; (ii) differences in the voltage levels due to different amounts of stored energy in the battery; (iii) different properties of the medium in different directions (anisotropic medium) and (iv) different interferences from neighboring nodes.

Given that protocol stacks tend to be implemented based on the assumption that unidirectional links do not exist, three techniques have been used to "hide" the unidirectional links: (i) *tunneling*; (ii) *blacklisting* and (iii) *transmission power increase*. If a link from node u to v is unidirectional, the tunneling approach attempts to find a path from v to u and give higher level protocols the illusion of a link from v to u. In order to achieve this, some routing functionality has to be performed at the lower layers of the protocol stack [19]. Packets sent across the tunnel have larger delays because they have to cross several hops. This is not too important though, because often the tunnel is used only for acknowledgements to packets that were sent across the unidirectional link. It is important however to avoid the *ACK explosion* [20]. Consider a unidirectional link from node N_u to node N_v. Consider also that there is a path from N_v to N_u. A data message has been sent across the link N_u to N_v and now the node N_v should send an ACK across the path back to N_u. However, the path from N_v to N_u contains a unidirectional link too. This link is from node N_x to N_y. When a packet has crossed the hop from

N_x to N_y, node N_y should send an ACK to N_x. In order to do this, it may have to find a path to N_x. It is possible that the path from N_y to N_x uses the link from N_u to N_v. This may generate an ACK from N_u to N_v and this process continues forever.

The technique of blacklisting detects unidirectional links when sending data messages, and does not use them in the future. The technique "hello" is similar but here “hello” messages are exchanged so a node i knows about the existence of a neighbor and whether they can hear i. This exchange is periodic and occurs regardless of whether the nodes are involved in routing data traffic or not. These techniques are sometimes called *ignoring* [21] or *check symmetry* [6]. Yet another technique to ignore unidirectional links is to treat it as a fault. This technique has been applied in conjunction with Ad-hoc On-Demand Distance Vector Routing (AODV) and it works as follows. When a source node attempts to find a route to the destination, it floods the network with Route-Request (RREQ) packets. In the normal AODV when RREQ packet reaches a node which knows a route to the destination, this node sends Route Reply (RREP) back on the same paths as the RREQ was sent on. With the normal AODV, RREP would fail on unidirectional links but instead this technique attempts to find a new path back to the source. When it finds a node with RREQ it knows a route back to the source node [22]. A similar scheme was proposed in [6] called *Bidirectional flooding*. Another technique (which we call “transmission power increase”) permits a downstream node of a unidirectional link to temporarily increase its power for sending responses such as acknowledgements and clear-to-send [23]. This technique is based on the sender to piggyback its geographical position obtained by GPS and the receiver should use this information to calculate the distance, which in turn is used to know how much the transmission power should be increased. We think the idea of increasing transmission power is interesting but in [23] the authors do neither give any details on how this increase transmission power is computed nor state the assumed path loss. Common to these techniques is that they require no or minimal changes to routing protocols.

Several routing algorithms have been proposed for unidirectional links. A common challenge that faces routing with unidirectional links is *knowledge asymmetry*; that is, if a link from u to v is unidirectional, only v can detect the existence of the link (by hearing a broadcast from u) but u is the one that will use the knowledge of the link for routing purposes. One technique builds on *distance vector*. The classic distance vector algorithm maintains a vector at each node and this vector stores the hop count to every other node N_i and the next node that should be used for forwarding to this node N_i (sometimes a sequence number is added too; it is used for updates).

Consider a node N_u with a neighbor N_v. Node N_v knows a route to node N_w. The number of hops from N_u to N_w is no larger than the number of hops from N_v to N_w plus one. If the link N_u to N_v is bidirectional this fact can be easily exploited in the design of a routing protocol because the length of the route N_v to N_w can simply be communicated over one hop to N_u. However, if the link N_u to N_v is unidirectional this is more challenging.

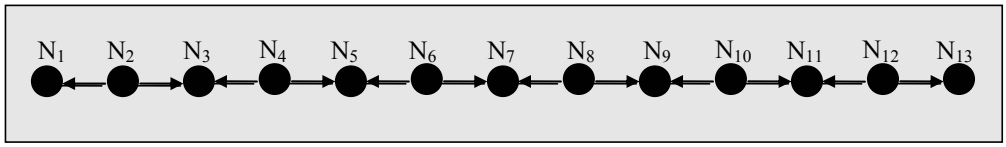
One extension of distance vector [21] however stores all distance vectors of all nodes in the network (hence it requires $O(m^2)$ storage). Another extension [24] sends information "downstream" until every node knows a circuit to itself. The node selects the shortest circuit and informs its upstream neighbors, and then the standard distance vector algorithm is used. Other techniques [19, 25] and [26] disseminate link state information across a limited number of hops. This is based on the assumption that the reverse path of a unidirectional link is short and this assumption has been supported empirically [19].

Pure link-state routing disseminates the topology information to all nodes and then the routes are calculated. This avoids the problem of asymmetric information (mentioned earlier) but the overhead of this scheme is large already.

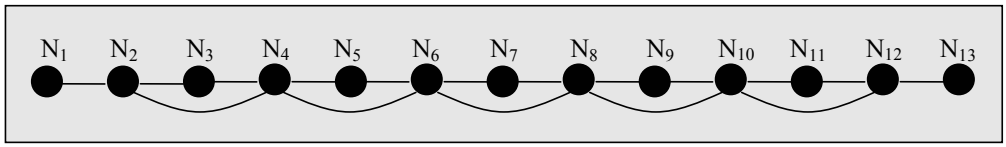
In order to reduce the routing cost in networks with unidirectional links, it has been suggested that a subset of nodes should be selected and only they should maintain routing information about all nodes in the network. It is required that all nodes which are not in this subset have a link from the subset and a link to the subset. Algorithms for selecting this subset of nodes have been proposed and they have very low overhead [27].

It has often been pointed out that unidirectional links should be avoided altogether because existing MAC protocols cannot deal with them (as we already mentioned, MACA, which was the basis for the RTS/CTS exchange in IEEE 802.11, relies on bidirectional links). But recently, this view has been challenged. For example [28] mentioned that their routing protocol works well for multicast and that it could be used for unicast routing as well – if there was a MAC protocol for unidirectional links.

To the best of our knowledge, the only previous MAC protocol that work for unidirectional links require synchronized clocks and it suffers from (an unbounded number of) collisions [8]. The technique in [8] addresses medium access control on unidirectional links. The technique generates pseudo-random numbers on each node and these numbers act as priorities. Every node knows the seed of the pseudo-random numbers on other nodes and hence a node knows if it has a higher priority than its neighbors. If it has, then it is the winner; otherwise it is not a winner. If it is a winner then it transmits in that time slot. Every new time slot, a new pseudo-random number is generated. This protocol is designed to deal with hidden nodes in the following



(a) Topology graph



(b) Interference graph

Fig. 7. An example of how the performance of our MAC protocol can be significantly improved if the topology is known.

way: if a node N_i has a neighbor with higher priority two hops way then node N_i simply does not transmit. This scheme is collision-free but it depends on synchronized clocks. Our protocol does not require that.

In the theory we assumed that $prop = 0$. We can easily extend the theory for the case when $prop > 0$. We can do it as follows. Select the time unit such that $(1-prop)$ is the time it takes to transmit a replica. Hence, if $prop = 1\mu s$ and the time to transmit a replica is 1 ms, then let 1.001 ms denote a time unit.

In this paper, we assumed topology is not known. However, if the topology is known we can perform significantly better (assuming that we also know the interference graph). Every node in the topology graph also exists in the interference graph. The links in the interference graph are non-directed. The links in the interference graph cannot simply be computed from the topology graph. However, there are some links in the interference graph that are necessary. Consider two nodes in the topology graph N_i and N_j. If there is a link from N_i to N_j or from N_j to N_i, then there is a link between N_i and N_j in the interference graph as well. If there is a node N_k with a link from N_i to N_k and a link from N_j to N_k then there is a link between N_i and N_j in the interference graph as well. From the interference graph, it is possible calculate Δ:s that cause a significant decrease in the overhead. This is illustrated in Example 2.

Example 2. Consider $m = 13$ nodes ordered in a line such that every node with an even index has two outgoing links; node i has a link to node i-1 and node i+1. This is illustrated in Figure 7a. If the topology is unknown, then we must assume that all 13 nodes can transmit simultaneously and can collide. A solution to (1),(2),(3) in Section 2.3 is the following Δ:s: $\Delta_1 = 22$, $\Delta_2 = 26$, $\Delta_3 = 34$, $\Delta_4 = 38$, $\Delta_5 = 46$, $\Delta_6 = 58$, $\Delta_7 = 62$, $\Delta_8 = 74$, $\Delta_9 = 82$, $\Delta_{10} = 86$, $\Delta_{11} = 94$, $\Delta_{12} = 106$, $\Delta_{13} = 118$ and $z = 1417$. From the interference graph shown in Figure 7b, we observe that every node has at most 4 links. This gives us $m = 5$, and we calculate the following Δ:s: 6, 10, 14, 22, 26. Now we can assign $\Delta_1 = 6$, $\Delta_2 = 10$, $\Delta_3 = 14$, $\Delta_4 = 22$, $\Delta_5 = 26$, $\Delta_6 = 6$, $\Delta_7 = 10$, $\Delta_8 = 14$, $\Delta_9 = 22$, $\Delta_{10} = 26$, $\Delta_{11} = 6$, $\Delta_{12} = 10$, $\Delta_{13} = 14$. Observe that we reuse Δ:s and this does not cause any collisions. In this way, we obtain $z = 105$, which is significantly lower. □

In general this requires solving the problem Achromatic Number which is known to be NP-hard (see page 191 in [29]) but several approximation algorithms are available. We can see from Figure 7 that z is unaffected by the size of the network; only the number of neighbors 2-hops away matters. Hence, this approach is efficient in large networks if they are not dense.

6 Conclusions

We have presented the first MAC protocol that can guarantee that the time from when an application requests to transmit until the message is transmitted is bounded even in the presence of unidirectional links and without using synchronized clocks or taking advantage of topology knowledge. We have implemented the protocol and observed that: (i) the effort required to implement it is small; (ii) by observing the number of lost messages we found that the implementation guaranteed that at least one replica of a message is collision-free and (iii) the number of lost messages at the receiver is significantly lower using our protocol than a replication scheme with random delays between replicas. We also run a scheme with random time for transmission with only one replica. We expect such a scheme to perform similarly to ALOHA [16], and we found that our protocol performed significantly better.

Acknowledgements

This work was partially funded by Fundação para Ciência e Tecnologia (FCT) and the Network of Excellence on Embedded Systems Design ARTIST2 (IST-004527).

References

[1] A. Cerpa, J. L. Wong, L. Kuang, M. Potkonjak, and D. Estrin, "Statistical Model of Lossy Links in Wireless Sensor Networks," in Information Processing in Sensor Networks (IPSN'05). Los Angeles, California, USA, 2005.

[2] A. Cerpa, N. Busek, and D. Estrin, "SCALE: A Tool for Simple Connectivity Assessment in Lossy Environments," UCLA Center for Embedded Network Sensing (CENS) Technical report 0021, September 2003.

[3] A. Woo, T. Tong, and D. Culler, "Taming the underlying challenges of reliable multihop routing in sensor networks," in Conference On Embedded Networked Sensor System. Los Angeles, California, USA, 2003.

[4] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker, "Complex Behavior at Scale: An Experimental Study of Low-Power Wireless Sensor Networks," UCLA Technical Report CSD-TR 02-0013, February 2002.

[5] D. Kotz, C. Newport, R. Gray, J. Liu, Y. Yuan, and C. Ellliot, "Experimental Evaluation of Wireless Simulation Assumptions," in International Workshop on Modelling Analysis and Simulation of Wireless and Mobile Systems, 2004.

[6] G. Zhou, T. He, S. Krishnamurthy, and J. Stankovic, "Impact of Radio Irregularities on Wireless Sensor Networks," in International Conference on Mobile Systems, Applications, and Services, 2004.

[7] J. Zhao and R. Govindan, "Understanding packet delivery performance in dense wireless sensor networks," in On Embedded Networked Sensor Systems,. Los Angeles, California, 2003.

[8] L. Bao and J. J. Garcia-Luna-Aceves, "Channel access scheduling in Ad Hoc networks with unidirectional links," in Workshop on Discrete Algothrithms and Methods for MOBILE Computing and Communications. Rome, Italy, 2001.

[9] "AMPL, www.ampl.com."

[10] "LOQO, http://www.princeton.edu/~rvdb/."

[11] B. Andersson, N. Pereira, and E. Tovar, "Delay-Bounded Medium Access for Unidirectional Wireless Links," Institute Polytechnic Porto, Porto, Portugal HURRAY-TR-060701, Available at http://www.hurray.isep.ipp.pt/asp/show_doc.asp?id=255, July 2006.

[12] A. Mok, "Fundamental Design Problems of Distributed Systems for the Hard Real-Time Environment," in Electrical Engineering and Computer Science. Cambridge, Massachusetts: Massachusetts Institute of Technology, 1983.

[13] "Crossbow, "MICAz - Wireless Measurement System Product Datasheet," 2005.

[14] J. Hill, "System Architecture for Wireless Sensor Networks," in Computer Science Department: University of California, Berkeley, 2003.

[15] A. Varga, OMNeT++ Discrete Event Simulation System. Tech. University of Budapest, Budapest, 2003.

[16] N. Abrahamson, "The ALOHA system - another alternative for computer communications," in 1970 fall joint computer communications, AFIPS Conference Proceedings. Montvale., 1970.

[17] P. Karn, "MACA - A New Channel Access Method for Packet Radio," presented at ARRL/CRRL Amateur Radio 9th Computer Networking Conference, 1990.

[18] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," in Mobile Computing, T. I. a. H. Korth, Ed.: Kluwer Academic Publishers, 1996.

[19] V. Ramasubramanian, R. Chandra, and D. Mossé, "Providing a Bidirectional Abstraction for Unidirectional Ad Hoc Networks," in IEEE INFOCOM. New York NY, 2002.

[20] S. Nesargi and R. Prakash, "A Tunneling Approach to Routing with Unidirectional Links in Mobile Ad-Hoc Networks," in IEEE International Conference on Computer Communications and Networks (ICCCN). Las Vegas, 2000.

[21] R. Prakash, "A routing algorithm for wireless ad hoc networks with unidirectional links," Wireless Networks, vol. 7, pp. 617 - 625, 2001.

[22] M. K. Marina and S. R. Das, "Routing performance in the presence of unidirectional links in multihop wireless networks," in 3rd ACM international symposium on Mobile ad hoc networking & computing. Lausanne, Switzerland, 2002.

[23] D. Kim, C.-K. Toh, and Y. Choi, "GAHA and GAPA : Two Link-level Approaches for Supporting Link Asymmetry in Mobile Ad Hoc Networks," IEICE Transaction on Communication, vol. E-86B, pp. 1297-1306, 2003.

[24] M. Gerla, L. Kleinrock, and Y. Afek, "A Distributed Routing Algorithm for Unidirectional Networks," in IEEE GLOBECOM. . 1983.

[25] L. Bao and J. J. Garcia-Luna-Aceves, "Unidirectional Link-State Routing with Propagation Control," in IEEE Mobile Multimedia Communications (MoMuC). Tokyo, Japan, 2000.

[26] T. Ernst, "Dynamic Routing in Networks with Unidirectional Links," INRIA, Sophia Antipolis 1997.

[27] J. Wu and H. Li, "Domination and Its Applications in Ad Hoc Wireless Networks with Unidirectional Links," in International Conference on Parallel Processing. Toronto, Ontario, Canada, 2000.

[28] M. Gerla, Y.-Z. Lee, J.-S. Park, and Y. Yi, "On Demand Multicast Routing with Unidirectional Links," in IEEE Wireless Communications & Networking Conference (WCNC). New Orleans, LA, USA, 2005.

[29] M. R. Garey and D. S. Johnson, Computers and Intractability A Guide to the Theory of NP-Completeness. New York: W. H. Freeman and Company, 1979.