# Collision Reduction in Random Access Slots for TDMA Tactical Mobile Ad Hoc Networks

Pascale FOUILLOT, Raphaël MASSIN, Catherine LAMY-BERGOT, Isabelle HERBIN, Gilbert MULTEDO
Thales Communications, 160, boulevard de Valmy Colombes, 92704, France, +33(0)1 41 30 30 00

**{pascale.fouillot, raphael-a.massin, catherine.lamy-bergot, isabelle.herbin, gilbert.multedo}@fr.thalesgroup.com**

## ABSTRACT

Ad hoc networking capabilities are quickly gaining in the domain of security, crisis management and military deployment. Modern network-centric warfare is in particular a good target for ad hoc networking approaches, as long as the different nodes of the network are able to properly maintain their local topology, i.e. one hop neighborhood, despite the channel impairments, possible jamming and network mobility. In this paper, a new contention access scheme for neighborhood maintenance in TDMA mobile ad hoc networks is proposed. Called CORIAS (COllision Reduction In Random Access Scheme) this algorithm has been designed to answer to typical military waveform TDMA definition, while integrating a collision control strategy minimizing the transmission delay for neighborhood maintenance signaling. Rationale and CORIAS detailed mechanisms are described in this paper, and simulation results illustrate its effectiveness.

## Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Architecture and Design – *wireless communciation;* Network protocols – *Aloha*

## General Terms

**Algorithms**.

## Keywords

Mobile ad hoc network, random access, TDMA, neighborhood management.

## 1. INTRODUCTION

A key point in the concept of modern network-centric warfare is that relevant information must be available in time where it is required. Flexible wireless communication solutions are needed to implement this in the battlefield environment, where the communicating nodes are mobile and where some of them may become out of order at any time. Ad hoc networking provides such a solution. In an ad hoc network all nodes are equal as regards network traffic forwarding: any single node may be source, destination and relay. Additionally any node may communicate directly with any other node in range of transmission. Communication with nodes out of transmission range is done along a multi radio hops path.

Numerous technical difficulties are associated with mobile ad hoc networking, and within the MANET IETF group a lot of research was conducted to define routing protocol for mobile ad hoc networks, such as AODV [1] and OLSR [2]. At the MAC level various access schemes have been studied and Time Division Multiple Access (TDMA) is considered today as a good technique to ensure high network capacity and to provide strong quality of service guarantees when targeted communication ranges are some kilometers long and latency constraints are reasonable. In TDMA, part of the bandwidth is dedicated to signaling transmission and the remainder is reserved for user data transmission. The way time slots belonging to the latter part are organized is either fixed or determined through the exchange of signaling messages over slots associated to the former part. Hopefully the bandwidth reserved for data is much larger than the one dedicated to signaling.

In both of the aforementioned routing protocols, HELLO messages are used so that each node discovers and maintains its local topology, i.e. knows when new nodes become one hop neighbors and when current one hop neighbors leave their transmission range. These protocols, primarily developed for wireless local area networks (WLAN), advocate a transmission frequency for HELLO messages which is on the order of one every few seconds. Nevertheless in high mobility conditions it is of critical importance to detect topology changes as soon as possible, in order to avoid data loss due to useless transmission to a node which is no longer a neighbor of the emitter. Therefore frequent local topology maintenance, established through HELLO message transmissions, is of paramount importance to the performance of a tactical mobile ad hoc network.

In TDMA context, HELLO messages are transmitted over signaling time slots. There are two options to allocate such time slots to nodes. First, one time slot can be statically associated to each node, through initial configuration. This means that only the owner of each signaling slot may use it for transmission. In that case, if the network is composed of $N_{max}$ nodes, one recurring signaling time slot out of $N_{max}$ will typically be associated to each node. When $N_{max}$ increases, the delay between two successive occurrences of a given node signaling time slot increases. A second option is to use an access strategy based on contention: any node may transmit during any signaling time slot, potentially causing collisions. This second strategy is not efficient in high density environment, but because it allows to benefit from potential spatial reuse, it is better than static allocation when the local node density is low to medium.

In tactical ad hoc networks the node density does not only vary spatially but also changes during a mission: high in the beginning when all nodes can be located in close proximity and are all in range of each other, low during operations when nodes are spread out on the operational theatre. Consequently, for such networks neither approaches always offer the most efficient solution, which led us to propose a solution adaptive to the local node density. This solution described hereafter is a contention access scheme for neighborhood maintenance in TDMA mobile ad hoc networks.

This paper is organized as follows. Section 2 introduces related works, Section 3 presents the system model. In Section 4 the principles of CORIAS are discussed while the details of the algorithm are described in Section 5. Simulation results are provided in Section 6 and finally Section 7 draws out conclusions and perspectives.

## 2. RELATED WORK

Radio access in presence of collisions is a topic that has triggered a lot of research and different solutions have been proposed.

Inherently, Aloha [3] and multi-access schemes derived from Aloha (such as WiFi DCF [4]) are contention based. In those schemes, when a node has some data to transmit and if it has determined through carrier sensing that the radio medium is available, then this node starts emitting data packets. As several nodes may transmit simultaneously in the same geographical area, collisions may happen. A node which has detected that a collision occurred during its last transmission waits during a random amount of time and then triggers a retransmission. The delay between successive transmissions failed due to collision increases exponentially. This is the exponential backoff scheme. This delay is expressed in slots. In WiFi [4] the duration of these slots depends on the physical layer technology but is no greater than 50µs (for frequency hopping spread spectrum physical layer). Such slots duration on the order of some tens of microseconds make it feasible to wait for tens of slots before attempting to transmit on the radio channel, because the waiting time duration is still shorter than one millisecond. In the ad hoc networks considered in this paper the slots duration are much longer, at least ten times longer. One reason is that possible communication ranges are longer (e.g. to compensate propagation delays for 30km transmissions, a 100µs guard time is required). Another reason is related to the fact that frequency bands are lower than standard civilian bands, which entails lower available bandwidth and longer transmission time for the same amount of transferred information. A third reason is that to increase the communications robustness to fading, frequency hopping is used within the same transmission, implying some additional guard time between successive dwells. Due to these reasons, in the tactical ad hoc networks considered here, the slots are some milliseconds long. The exponential backoff scheme which is the usual strategy applied in contention access schemes becomes extremely inefficient in the TDMA networks considered in this paper, and is not usable.

In 3GPP there are two uplink random access transport channels: RACH and CPCH [5]. The associated procedures take advantage of the asymmetric nature of communication in cellular networks, i.e. upon reception of a message on the RACH or CPCH a base station promptly sends ACK/NACK to the transmitting UE (User Equipment), on a collision free channel. In ad hoc networks, no similar asymmetry exists and this approach is not possible.

These major differences intrinsic to tactical ad hoc networks prompted us to investigate other domains confronted with collision resolution issues. The RFID (Radio Frequency IDentification) networks belong to such a domain where a lot of communicating nodes must quickly identify each other. A typical RFID system is made up of a lot of RFID tags and one RFID reader that collects data stored in the tags attached to physical items. One key technical issue is how to resolve the collisions caused by a reader and multiple tags trying to transmit commands and data simultaneously through a shared wireless communication medium. Different solutions, based on the division of tags in smaller groups, have been proposed [6][7] to this difficulty. Within each group, tags use a slotted Aloha strategy to access the radio channel. These solutions can not be directly applied in ad hoc networks for in such networks, all nodes are both emitter and receiver. It is, then, not possible to clearly identify if nodes could be compared to RFID readers or if nodes could be compared to RFID tags. Nevertheless, some key principles can be adapted to the mobile ad hoc context, leading to the CORIAS strategy.

## 3. CONSIDERED SYSTEM MODEL

As already stated, we consider a TDMA access scheme. Time synchronization is considered maintained through other means (e.g. use of GNSS) TDMA means that time is divided in slots. As illustrated in Figure 1, these slots are grouped in what is called a TDMA frame. Within each TDMA frame there are n slot reserved for signaling and a number (not specified here as we focus on the signaling part) of slots reserved for data transmission. More precisely signaling slots are random access slots (RAS) to be used to transmit HELLO messages. Moreover only one common signaling channel is used. When one node NODE transmits during a RAS, all nodes within communication range which are not also transmitting may receive the transmission from NODE. HELLO messages are supposed to be transmitted using maximum available power and the most robust available modulation and coding scheme.

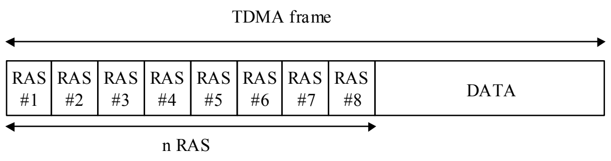


**Figure 1 TDMA frame format.**

The network consists of $N_{max}$ nodes $S_i$, with i in { 1, 2, …, $N_{max}$}. A one hop neighbor of $S_i$ is a node which is in communication range of $S_i$. At time t, each node $S_i$ has $N_i(t)$ one hop neighbors. To simplify notations, $N_i(t)$ is written $N_i$ in the remainder of the paper. All nodes are assumed to transmit regularly in each frame if the channel access strategy does not specify any restriction. In such a configuration, a node $S_i$ has $N_i$ nodes attending to access the channel per frame.

## 4. CORIAS PRINCIPLES

In line with the typical use and considered system model mentioned in the previous section, CORIAS goals are to:

1. minimize the delay between two successive HELLO message successful transmissions between any node and its one hop neighbors,
2. guarantee that HELLO messages collision probability will always be below a threshold $P_{collision}$ whose value is chosen through configuration,
3. achieve the two preceding goals whatever the local node density.

Achieving these goals actually mean improving in a fundamental way the network reactivity with respect to the node(s) mobility, as will be illustrated in the numerical results given in Section 6.

Let us detail better the proposed approach and for this consider what happens with a "naïve" random access scheme. In this scheme, called RANDOM, all nodes are allowed to randomly choose one RAS per frame to transmit one HELLO message. With RANDOM the probability of collision depends on two parameters: the number of neighbors and the occurrence of RAS. Assuming the system model introduced in section 3, a node $S_i$ has $N_i$ (subsequently written N to simplify notations) neighbors, and assuming a uniform probability of RAS selection, this node has a probability of choosing a given RAS equals to 1/n. The probability of good reception $P_{rx} = 1 - P_{collision}$, that no other node of the N neighbors chooses the same slot is thus:

$$P_{rx} = \left(1 - \frac{1}{n}\right)^{N-1}. \quad (1)$$

As a consequence, the probability of collision is:

$$P_{collision} = 1 - \left(1 - \frac{1}{n}\right)^{N-1}. \quad (2)$$

Equation 2 highlights that in such a configuration, the collision probability depends on the number of nodes that intend to access the channel and the number of RAS that are available in a given frame. In order to reduce collisions in a network, one can tune these two parameters. As already mentioned, CORIAS is inspired by the RFID concept which adjusts the number of nodes allowed to access the channel in the same frame. In order to do so, a node which has more than a few neighbors requests them to split into different groups, each group being then allowed to transmit only on a subset of the frame.

In the RFID context, a network is composed of one RFID reader (receiver) and several tags (emitter) that need to be identified by the RFID reader. In such a configuration, the reader is the only equipment listening to the other ones to obtain information, the tags listening only for signalization and for commands made to ensure an efficient transmission when a reader is close to the tag. As detailed in [6][7], one can use a procedure to reduce collisions in this RFID context, in which the receiver estimates the number of emitters that intend to access the channel (number of neighbors). When the number of estimated neighbors leads to a collision probability higher than a given threshold, the receiver requests its neighbors to split into groups via the transmission of a constraint in a reserved slot. In that scheme, it is thus up to the receiver to manage the neighborhood.

In ad hoc networks, all nodes can be either emitter or receiver, which implies that all nodes must manage their own neighborhood. This first leads to operate both transmission (TX) and reception (RX) parts of the algorithm on each node, and second to distribute the algorithm over the different nodes. It is then up to each node as receiver to manage its own neighborhood and up to each node as emitter to submit to the possibly different constraints that can be received.

Another required adaptation corresponds to the efficiency of the algorithm. In the RFID context since only one node is sending constraints, a slot can be reserved to the RFID reader without wasting exaggeratedly resources. On the contrary, in ad hoc networks, since all nodes send constraints, reserving a slot to each node of the network would result in a static allocation which does not match with the necessity of having a high opportunity of transmission even in networks composed of a large number of nodes. To solve this problem, in CORIAS we propose to send the constraints on RAS, within HELLO messages. Even if using RAS for constraint transmission may first appear risky, the feasibility of the collision reduction process relies on the fact that the algorithm used to determine the constraint is an iterative process. To determine the constraint, each node, as a receiver, estimates the number of neighbors intending to access the channel. The uncertainty of such an estimation comes from the unknown number of nodes in collision. At the beginning of the process, if many collisions occur the estimation is far from the reality. The constraint computed from this estimation may consequently be different from the ideal one. But as soon as a constraint is computed and received by neighbors, neighbors split into different groups leading to less collisions in the next frame. The reduction of collisions implies an improvement of the neighbor estimation, leading to a more adapted constraint, in an iterative process illustrated by Figure 2. As shown in the numerical results, the needed number of iterations to converge is fairly low.

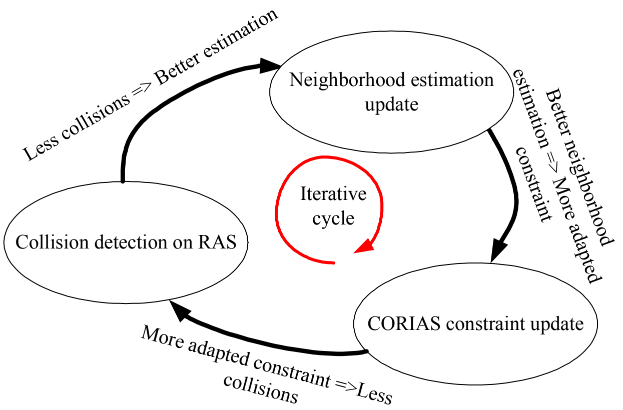

Less collisions => Better estimation
Neighborhood estimation update
Better neighborhood estimation => More adapted constraint
Iterative cycle
Collision detection on RAS
CORIAS constraint update
More adapted constraint =>Less collisions

**Figure 2 Iterative process principle.**

## 5. CORIAS ALGORITHM

As in the RFID context, the main phases of the algorithm are: neighborhood estimation, constraint generation and management, and constraint application. However in our case the nodes of the network are responsible for managing their own neighborhood, and for implementing adequate actions to adapt to their neighbors constraints.

The following subsections detail each of these three main phases, and a way to improve and stabilize the algorithm is presented in a fourth sub-section.

### 5.1 Neighborhood estimation

Assuming that all neighbors of a given node S transmit one message in each frame, the node S is able to estimate its neighborhood only considering the state of each slot. The different slots of the frame are readable, idle or collided. A readable slot implies that one neighbor accessed the channel while an idle slot conveys that no node intended to access the channel and a collided slot means that at least two nodes accessed the channel. Let nbReadable and nbCollided respectively be the number of readable and collided slots detected in a frame, the node S can thus estimate the number of neighbors using Equation 3.

$$N_{estim} = nb\mathrm{Re}adable + k * nbCollided \tag{3}$$

Note that this calculus is only an estimation due to the unknown number of neighbors that collide in a slot. Naturally, setting a k coefficient too high would imply unnecessary divisions in groups and consequently slot usage losses, so it must be chosen properly. In CORIAS, k is set during the system initialization phase, taking into account the ratio of the average number of neighbors nodes have in the network over the number of RAS in the frame. The higher this ratio is, the higher the k factor should be.

### 5.2 Constraint generation and transmission

Once the considered node S has estimated the number of neighbors it has, it can compute the constraint to send to its neighbors to reduce the collision probability. The non collision probability at node S evolves as described in Equation 1.

Consequently, in order to get a probability of collision smaller than a threshold $P_{threshold}$, the number of neighbors intending to communicate in a frame must not be higher than M whose value is calculated as follows:

$$M = 1 + \ln\left(P_{threshold}\right) \Big/ \ln\left(1 - \frac{1}{n}\right) \tag{4}$$

If the number of estimated neighbors is higher than the threshold M, then the considered node must ask its neighbors to split up in a number Q of different groups. The constraint Q is calculated using Equation 5 and is included within the next HELLO messages sent by the considered node.

$$Q = \left\lfloor N_{estim} \big/ M \right\rfloor + 1 \cdot \tag{5}$$

where $\lfloor X \rfloor$ represents the integer part of X.

The Q value is the only field added to HELLO messages required to operate CORIAS. It is 4 bits long.

Remains for each node to apply the constraint in a manner such as to ensure that the Q groups are of equivalent size. In CORIAS, it is proposed to split in Q groups based on the nodes MAC addresses, assuming a uniform MAC address sharing around the node S. All nodes whose MAC address modulo Q equals to x belong to group x+1 and are allowed to transmit in frames whose identifier modulo Q equals x.

Since the neighbors receiving the constraint split in Q groups, Q frames are required for all neighbors to transmit a message to node S. The node S which sends the constraint Q must therefore estimate its neighborhood on Q frames. Whereas in Equation 3 the estimation and constraint update transmission where done each frame, one operates now every Q frames. Denoting nbReadable(t) and nbCollided(t) respectively the number of readable and collided slots detected in frame t, the number of estimated neighbors becomes:

$$N_{estim} = \sum_{t=1}^{Q} (nbReadable(t) + k * nbCollided(t)) \text{ with } k \geq 2 \qquad (6)$$

Figure 3 presents an example of constraint update for a node S having derived a constraint Q=3 at the end of a frame 10.

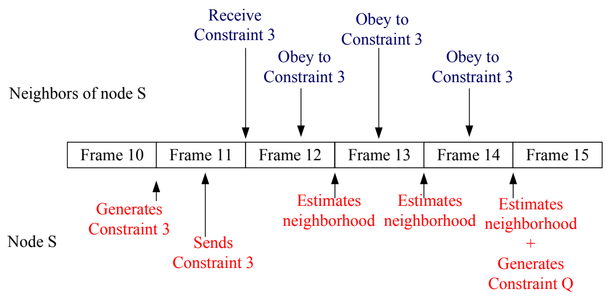


**Figure 3 Example of constraint update.**

In that example, the node S which computes the constraint Q=3 at the end of frame 10 sends it in frame 11. The constraint is taken into account by neighbors from frame 12. Node S can thus re-estimate its neighborhood and adjust its constraint based on information from frame 12 to 14.

Since the algorithm is distributed on all nodes of the ad hoc network, a node S itself has to obey a constraint q imposed by its own neighbors. In these conditions, node S which has just generated a constraint Q in a frame t might not be allowed to transmit in the frame t+1. Let v be the frame in which node S is allowed to transmit. Thus the node S must report its neighborhood estimation to frame v+Q, based on local estimation done during frame v+1 to v+Q. CORIAS adapts to this possibility of non emission by keeping the constraints previously received while a new constraint is not received as it will be described in section 5.3.

Figure 4 provides an example for a node S with a MAC address equals to 6, which has to submit to constraint q=2 and which computed a constraint Q=3 at the end of frame number 10.

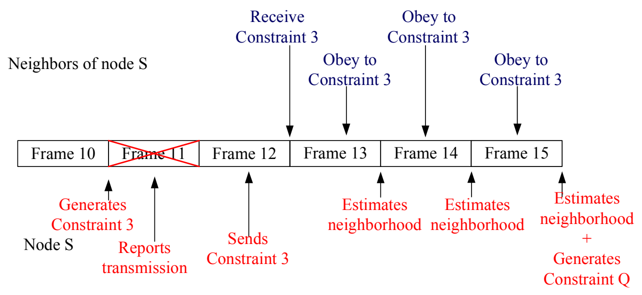


**Figure 4 Example of constraint update.**

Since node #6 obeys to constraint q=2, it cannot transmit in frame #11. It must thus report its transmission containing the new constraint Q=3 that has just been generated to frame #12. Its new neighborhood estimation must consequently start on frame #13. Since node #6 sends constraint Q=3, the estimation will finish at the end of frame #15.

As shown in the examples, some of the slots are not used for the estimation. In particular slots during which the constraint is sent and slots during which the transmission is reported. In order to avoid loss of information, CORIAS specifies a process to take these slots into account, and adapts to the possibility of deriving several neighborhood estimates by averaging them.

More precisely, neighbors keep obeying to the constraint previously received in case of non reception of a constraint. During slots where the constraint is sent and slots where the transmission is reported, the constraint to which neighbors submits to remains the same as the one to which they obeyed in precedent frames. As a consequence, nodes transmitting in frames, which numbers modulo the constraint are equal, remain the same (i.e neighbors transmitting in a given frame t are the same as in frame t+Q). As a result, the neighborhood estimation using frames t to t+Q-1 leads to the same result as the one using frames t+1 to t+Q (provided that no collision occurred). A node sending a constraint Q to its neighbors can consequently estimates its total neighborhood at the end of each frame after Q frames.

In the Figure 4 example, let us assume that node S has generated constraint Q=2 at the end of frame #7. the constraint is sent during frame #8. Node S estimates its neighborhood in frames #9 and #10. Since neighbors only receive the constraint Q=3 at the end of frame #12, they keep using constraint Q=2 during frame #11 and #12. Thus, neighbors that sent messages in frame #11 are the same than neighbors which sent messages in frame #9. A new estimation is possible based on frame #10 and #11. In the same way, a third estimation is possible based on frame #11 and #12. In order to take into account all the neighborhood estimations, results are weighted with older estimations using Equation 7.

$$N_{estim} = \alpha . N_{estim} + (1 - \alpha) . N_{lastEstim} \qquad (7)$$

Where $N_{lastEstim}$ is the last estimation of the neighborhood and α a weight coefficient bounded by 0 and 1 which can be set as an example equal to 0.8.

Thanks to this mechanism, all frames are used to estimate the neighborhood. Moreover, it smoothes the estimation, taking into account possible neighborhood evolution due to mobility, and consequently avoids strong variations in the constraints generated by nodes.

## 5.3 Constraint application

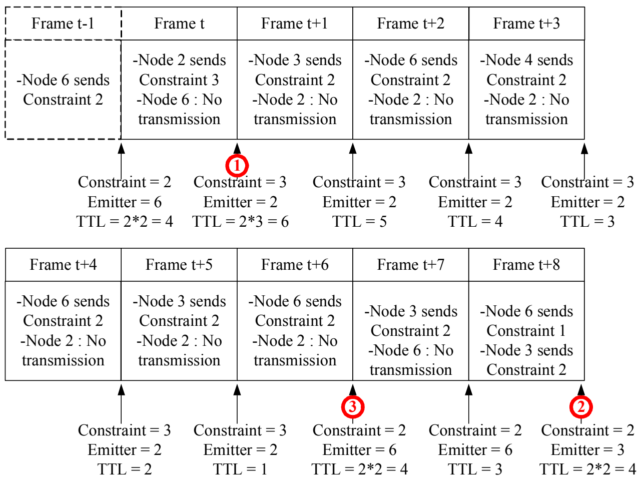
Since the algorithm is distributed on all nodes, a node having N neighbors receives N constraints. In order to guarantee that all nodes have at least a probability of collision smaller than $P_{collision}$, a node must submit to the highest constraint it receives from its neighborhood.

As the neighbors also have to obey to different constraints, they might not all send their constraints in the same frame (and a node is anyway unable to receive several messages in the same timeslot). As a consequence, the considered node must not update at each frame the constraint q to which it submits to. Otherwise there would not be any guarantee that the highest constraint computed by its whole neighborhood is taken into account. To guaranty this rule, a node currently obeying to a constraint q announced by one of its neighbors i submits to a new received constraint $q_r$ only if:

1. $q_r$ is higher than q,
2. or $q_r$ is sent by node i,
3. or node i has not sent any constraint for a long time.

The first rule ensures the modification of the constraint value if a neighbor increases its constraint due to an increase of its own neighborhood or if a new node imposing a higher constraint enters in the neighborhood of the considered node. The second rule makes sure that the constraint is updated if the neighbor to which the considered node submits to reduces its constraint due to a reduction of its own neighborhood. Finally, the third rule ensures that the constraint is updated if the neighbor imposing the constraint to the considered node submits has left the neighborhood. However this last rule has a drawback: should this neighbor still be present but its constraint messages be lost in transmission, the reduction of constraint may lead to more collisions. This led us to introduce the corrective mechanisms detailed in sub-section 5.4. In order to apply the third rule, each node of the ad hoc network uses a Time-To-Live (TTL) counter. Each time the constraint q is updated, the TTL counter is set to 2*q. This counter is reduced by one at the end of a frame during which the constraint was not updated. If the counter reaches zero, then it is considered that the node imposing the constraint has left the neighborhood.

Figure 5 presents an example of received constraint update. The (red) surround numbers correspond to the different rules that trigger a constraint update. The first rule is applied at the end of frame t, while rules 3 and 2 are respectively applied at frame t+6 and t+8.


Frame t-1
-Node 6 sends Constraint 2
Frame t
-Node 2 sends Constraint 3 -Node 6 : No transmission
Frame t+1
-Node 3 sends Constraint 2 -Node 2 : No transmission
Frame t+2
-Node 6 sends Constraint 2 -Node 2 : No transmission
Frame t+3
-Node 4 sends Constraint 2 -Node 2 : No transmission
Constraint = 2 Emitter = 6 TTL = 2*2 = 4
1
Constraint = 3 Emitter = 2 TTL = 2*3 = 6
Constraint = 3 Emitter = 2 TTL = 5
Constraint = 3 Emitter = 2 TTL = 4
Constraint = 3 Emitter = 2 TTL = 3
Frame t+4
-Node 6 sends Constraint 2 -Node 2 : No transmission
Frame t+5
-Node 3 sends Constraint 2 -Node 2 : No transmission
Frame t+6
-Node 6 sends Constraint 2 -Node 2 : No transmission
Frame t+7
-Node 3 sends Constraint 2 -Node 6 : No transmission
Frame t+8
-Node 6 sends Constraint 1 -Node 3 sends Constraint 2
Constraint = 3 Emitter = 2 TTL = 2
Constraint = 3 Emitter = 2 TTL = 1
3
Constraint = 2 Emitter = 6 TTL = 2*2 = 4
Constraint = 2 Emitter = 6 TTL = 3
2
Constraint = 2 Emitter = 3 TTL = 2*2 = 4
**Figure 5 Example of received constraint update.**

## 5.4 Adaptations to collisions during constraint transmission

Because of remaining collisions, nodes may not receive some constraints. When the missed constraint does not correspond to the highest constraint sent by neighbors, the collision has no harmful consequence on the algorithm. But if the collision happens on the message conveying the highest constraint in the neighborhood and if the TTL counter reaches the zero value, the considered node may obey to a smaller constraint. Such an event has two main consequences. It implies first more collisions and second problems in the constraint generation. Indeed, the considered node obeying to a smaller constraint transmits more messages than expected by the neighbor which sent the highest constraint. The latest thus overvalues the number of neighbors which can cause an increase in the Q constraint.

Figure 6 to Figure 8 present un example of this phenomenon. Let us assume an ad hoc network as presented in Figure 6. The constraints represented on the figure correspond to each constraint generated by each node of the network at the end of frame #11. Before frame #11, all nodes obey to constraint q=1.

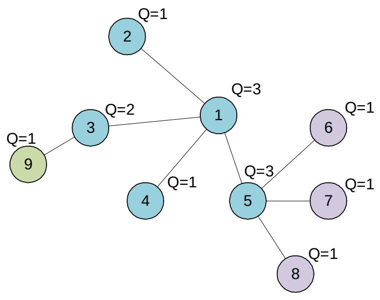


**Figure 6 Network configuration.**

During frame #12, all nodes transmit a HELLO message with the constraint they have previously generated. The Figure 7 presents node #1 neighbors transmissions.

| Frame #12 | | | | |
|---|---|---|---|---|
| | Slot 1 | Slot 2 | Slot 3 | Slot 4 |
| Node #2 | Tx Q=1 | | | |
| Node #3 | | | | Tx Q=2 |
| Node #4 | | Tx Q=1 | | |
| Node #5 | | Tx Q=3 | | |

**Figure 7 Constraint non-reception.**

In the example, a collision happens on slot #2. Thus, node #1 does not receive the constraint Q=3 from node #5 and obeys to Q=2 sent by node #3. Node #5 which imposes the constraint Q=3 to its neighbors estimates its neighborhood in frame #15. Figure 8 presents the transmissions in the neighborhood of node #5 during frame #13,#14, #15.

| | Frame #13 | Frame #14 | Frame #15 |
|---|---|---|---|
| Node #1 | Obeys to q=2 Tx | Obeys to q=2 | Obeys to q=2 Tx |
| Node #6 | Obeys to q=3 | Obeys to q=3 | Obeys to q=3 Tx |
| Node #7 | Obeys to q=3 Tx | Obeys to q=3 | Obeys to q=3 |
| Node #8 | Obeys to q=3 | Obeys to q=3 Tx | Obeys to q=3 |

**Figure 8 Consequence of constraint non-reception.**

Since node #1 obeys to Q=2, it transmits two times in frames #13 to #15. Assuming no collision in the frame, node #5 estimates 5 nodes in its neighborhood at the end of frame 15 which can cause an increase of its transmitted constraint.

This phenomenon is limited in time since collisions are limited. The constraint will be received anew and the equilibrium will be found once more. Even if this phenomenon corrects itself, CORIAS provides two solutions which allow to reduce the number of occurrences of un-wished constraint changes and limit the convergence time to find again the most appropriate constraint and limit collisions.

The two solutions rely on the fact that nodes obey to the highest constraint they received from their neighborhood (see section 5.3). Let us consider a node S with N neighbors. If all constraints are properly received by S, this latest never obeys to a constraint smaller than the constraint Q imposed by a neighbor, whatever the neighbor of S considered. This also means that if no collision disrupts constraint reception on node S, this latest never transmits more than one message in Q frames, with Q the constraint sent by a neighbor of S, whatever the neighbor considered.

In the first solution, CORIAS adds another field in HELLO messages. On top of the constraint $Q_S$ a node S imposes to its neighbors, the constraint $q_S$ to which the node obeys is sent in HELLO messages. When receiving the message, a considered neighbor T can detect a wrong constraint submission, if the value of $q_S$ is smaller than the constraint $Q_T$ sent by the considered neighbor itself.

The second solution consists in using the MAC address of emitters to check if nodes properly received the constraint. If a node S obeys to a constraint higher or equal to the constraint send by one of its neighbors T, it never transmits more than one message per $Q_T$ frames. If this condition is not respected, it means that the node obeys to a weaker constraint (and thus transmits more often than what it should do) due to non-reception of the right constraint.

When errors are detected, nodes can take them into account in their next neighborhood estimation. The estimation can either be reported, waiting for a new cycle where no error occur, or directly corrected.

These two solutions are complementary. They provides different advantages and can be both used together. On one hand, the first solution presents the advantage of allowing nodes to detect more errors than the second solution. Indeed, the reception of only one message containing the wrong constraint is sufficient to detect the error when at least two messages are needed to detect that a node has transmitted two times in a cycle in the second solution. On the other hand, the second solution allows nodes to correct neighborhood estimation when errors are detected whereas it is not possible using the first solution.

Theses two solutions avoid most of un-wished constraint changes. The only un-wished changes remaining happened when a first collision occurs avoiding the reception of the right constraint and a second collision occurs when using the wrong constraint. The second collision prevents nodes from detecting the error since neither the MAC address nor the constraint q to which the node obeys are received. This explain some few and brief constraint changes in the simulation results shown in section 6.

## 6. SIMULATION RESULTS

### 6.1 Simulation parameters and metrics

To assess the performance of CORIAS, simulations have been made using a simple custom event-driven discrete time network simulator. In this simulator an instance of the distributed CORIAS algorithm is run on each node. The network topology is determined through a trajectory file which provides the (x,y) coordinates of each node along time. If the distance between two nodes is no greater than a constant distance, these nodes are considered neighbors and can exchange messages. Non-neighboring nodes are not considered in range. As detailed in section 3, HELLO messages are periodically exchanged between neighboring nodes on RAS slots. If one node simultaneously receives more than one HELLO message, this is considered a collision and all messages are lost.

The value of parameter k used during neighborhood estimation is chosen as 2 and the weighted coefficient α is set to 0.80. Also, only the first mechanism (section 5.4) to reduce the impact of collision during one node's constraint transmission has been implemented.

The first performance metric considered in CORIAS simulations is the measured good reception rate $p_{rx}$ compared to $P_{threshold}$, i.e. the number of good receptions divided by the total number of receptions. For this metric CORIAS is only compared to the RANDOM algorithm (section 4) due to space constraint. The two other considered metrics are specific to CORIAS and measure (1) the time required so that the constraint to which a node submits has not changed during the last 10 TDMA frames and (2) the percentage of time when a node transmit an incorrect constraint.

Two different types of deployment, corresponding to low and medium density are considered hereafter. They were built to test various neighborhood configurations within a limited set of trials, in order to validate by simulation the interest of CORIAS.

## 6.2 Low density deployment

First the 13-nodes network of Figure 9 has been considered. In this network the nodes neighbors number is between one and four. The target probability of good reception $P_{thresholdx}$ value is 88%. In that scenario the simulation time is 1800 TDMA frames long. The number n of RAS in one TDMA frame is 10.

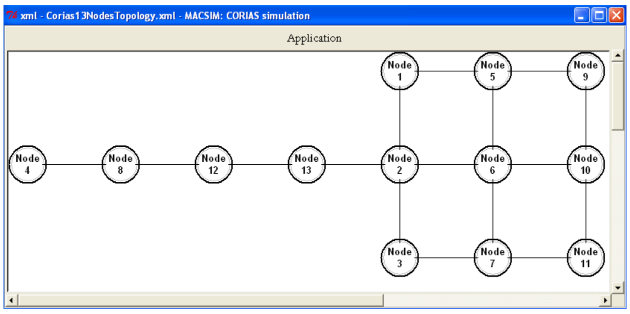


**Figure 9 Topology for low density scenario.**

### *6.2.1 Static deployment*

Initially, all nodes are static. Table 1 provides for each node the measured good reception rates for both RANDOM and CORIAS. As soon as the number of neighbors becomes greater than 2, with RANDOM the measured reception rate $p_{rx}$ is smaller than $P_{threshold}$. With CORIAS the target reception rate is always reached.

**Table 1 Good reception rates for low density scenario**

| Node | Neighbors | Good reception rate | |
|---|---|---|---|
| | | RANDOM | CORIAS |
| 4 | 1 | 1.00 | 1.00 |
| 1 | 2 | 0.89 | 1.00 |
| 3 | | 0.90 | 1.00 |
| 8 | | 0.91 | 0.90 |
| 9 | | 0.90 | 1.00 |
| 11 | | 0.90 | 1.00 |
| 12 | | 0.89 | 0.95 |
| 13 | | 0.90 | 0.94 |
| 5 | 3 | *0.81* | 0.95 |
| 7 | | *0.81* | 0.95 |
| 10 | | *0.80* | 0.95 |
| 2 | 4 | *0.74* | 0.90 |
| 6 | | *0.73* | 0.90 |

Table 2 provides further insight in the operation of CORIAS regarding the values of transmitted constraint Q and highest received constraint q, the amount of time required for each node to find the correct value of the constraint Q to transmit, and also the percentage of time during which each node transmitted an incorrect constraint Q. For nodes with only one or two neighbors, the required value for Q is 1 meaning that the initial value is already the final one (convergence time = 0). For nodes having more than two neighbors, this time (depending on k and α coefficients) remains small (convergence time < 10 TDMA frames), demonstrating that CORIAS is able to quickly reach the proper state.

**Table 2 CORIAS metrics for low density scenario**

| Node id | Neighbors number | Sent constraint (Q) | Highest received constraint (q) | Convergence time (TDMA frames) | % time with bad sent constraint |
|---|---|---|---|---|---|
| 4 | 1 | 1 | 1 | 0 | 0.00 |
| 1 | 2 | 1 | 2 | 0 | |
| 3 | | 1 | 2 | 0 | |
| 8 | | 1 | 1 | 0 | |
| 9 | | 1 | 2 | 0 | |
| 11 | | 1 | 2 | 0 | |
| 12 | | 1 | 1 | 0 | |
| 13 | | 1 | 2 | 0 | |
| 5 | 3 | 2 | 2 | 8 | |
| 7 | | 2 | 2 | 8 | |
| 10 | | 2 | 2 | 7 | |
| 2 | 4 | 2 | 2 | 7 | |
| 6 | | 2 | 2 | 5 | 0.22 |

Nevertheless, as discussed in section 5.4, sometimes the value of constraint Q transmitted by a node may be incorrect during a short duration before returning to its correct value. This case happens once on node 6 during the simulation, as highlighted in Figure 10. Around time 1300 several successive collisions occurred in the vicinity of node 6. Consequently some of its neighbors obeyed to a lower constraint and transmitted more often than expected by node 6. This induced an error in the neighbor count estimation of node 6 causing an increase in its announced constraint Q. This transient error has quickly been fixed as visible in Figure 10, demonstrating the stability of CORIAS mechanisms.

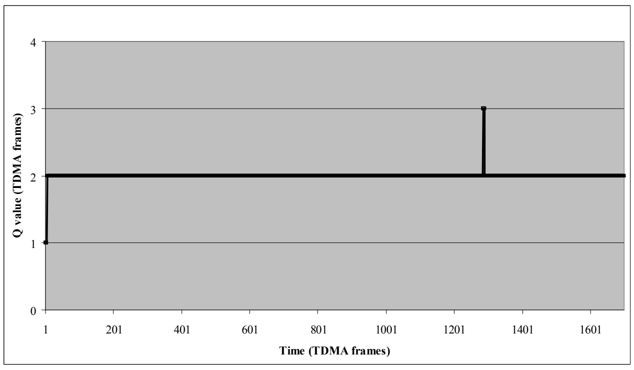


**Figure 10 Variation of Q value on node 6.**

### *6.2.2 One node mobility*

To confirm CORIAS relevance in presence of mobility, node 4 now moves horizontally from its position in Figure 9 along the {node 8, node 10} axis. Such a scenario can occur with aerial communication nodes and presents the advantage to limit the complexity of constraint evolution analysis. Some nodes change their transmitted constraint Q when a node previously out of range becomes neighbor. Figure 11 reports how node 6 updates its transmitted constraint Q during the simulation, regarding its number of neighbors.

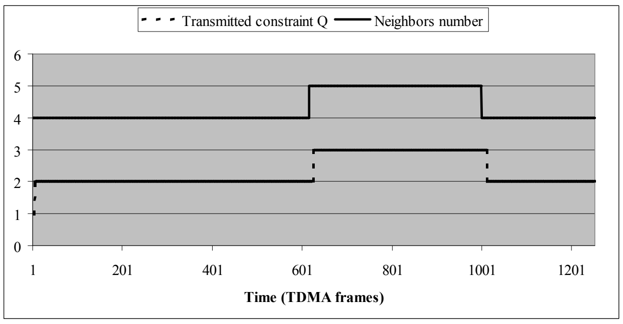


**Figure 11 Node 6 constraint and neighbors number variation.**

.

Table 3 provides the good reception rates averaged during the whole simulation for both RANDOM and CORIAS. Using RANDOM, the good reception rates for all nodes, except node 8, remain equal or becomes lower than those obtained in the static deployment. The reason of this phenomenon is that the mobility implies a temporary increase of the number of neighbors of all nodes except for node 8 which is the only one for which mobility implies a reduction of its number of neighbors. Using CORIAS, except for node 2, the target reception rate ($P_{threshold} = 88\%$) is always reached. The reason why node 2 did not reach the target rate is related to the way node identifiers are geographically spread out. Node 2 neighbors are nodes 1, 3, 6 and 13. When node 4 became one of its neighbors, node 2 increased its constraint Q from 2 to 3. As mod(1,3) = mod(4,3) = mod(13,3) and mod(3,3) = mod(6,3), the neighbor set of node 4 was only split in 2 groups instead of 3, one subset of slots being in practice never used, leading to more collisions than expected. This phenomenon can be avoided if nodes compute the constraint taking into account the distribution of their neighbors in groups on top of the neighborhood estimation. (e.g: if a node detects that none of its neighbors has sent a message in a given frame, it will upgrade its constraint by one to obtain a better distribution of its neighbors).

**Table 3 Good reception rates with mobility**

| Node identifier | Neighbors number | Good reception rate | |
|---|---|---|---|
| | | RANDOM | CORIAS |
| 8 | 1 to 2 | 0.96 | 0.98 |
| 4 | 1 to 5 | 0.86 | 0.93 |
| 1 | 2 to 3 | 0.89 | 0.97 |
| 3 | | 0.90 | 0.99 |
| 9 | | 0.89 | 0.98 |
| 11 | | 0.89 | 0.94 |
| 12 | | *0.85* | 0.95 |
| 13 | | 0.87 | 0.93 |
| 5 | 3 to 4 | *0.80* | 0.94 |
| 7 | | *0.79* | 0.94 |
| 10 | | *0.78* | 0.93 |
| 2 | 4 to 5 | *0.70* | *0.87* |
| 6 | | *0.70* | 0.89 |

## 6.3 Medium density deployment

In a second scenario, the 16-nodes meshed network of Figure 12 has been considered. In this network all nodes have 15 neighbors. The number n of RAS in one TDMA frame is 8. The target probability of good reception $P_{threshold}$ value is 80%, implying a Q constraint equal to 6. The simulation time is 1000 TDMA frames long.

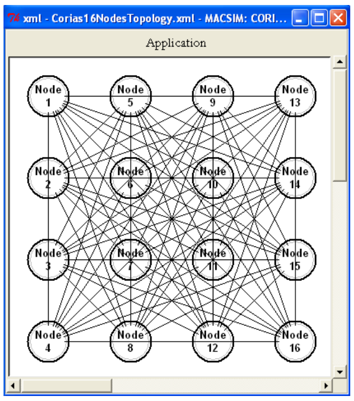


**Figure 12 Topology for medium density scenario.**

In this scenario, the RANDOM and the CORIAS algorithm are once more compared as regards the good reception rate metric, averaged on all nodes. Because RANDOM does not control collisions, the good reception rate value is very low: 0.16. Conversely, with CORIAS it is 0.81, higher than $P_{threshold}$. Table 4 provides the per node good reception rate values.

**Table 4 Good reception rates in medium density**

| Node id. | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| RANDOM | *0.15* | *0.15* | *0.15* | *0.15* | *0.15* | *0.15* | *0.15* | *0.15* |
| CORIAS | 0.81 | 0.80 | 0.81 | 0.81 | 0.81 | 0.80 | 0.81 | 0.80 |
| Node id. | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| RANDOM | *0.15* | *0.15* | *0.18* | *0.18* | *0.18* | *0.18* | *0.15* | *0.15* |
| CORIAS | 0.81 | 0.81 | 0.82 | 0.82 | 0.82 | 0.82 | 0.81 | 0.81 |

The topology being symmetrical all nodes exhibit the same behavior. As examples Figure 13 and Figure 14 report the good reception rates measured on node 6 during the simulation using respectively RANDOM and CORIAS algorithms.

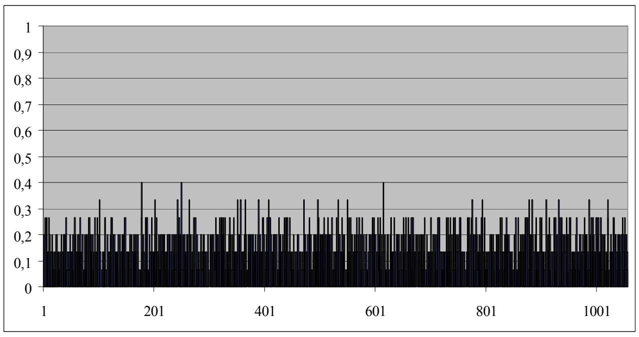


**Figure 13 Good reception rate for node 2 with RANDOM.**

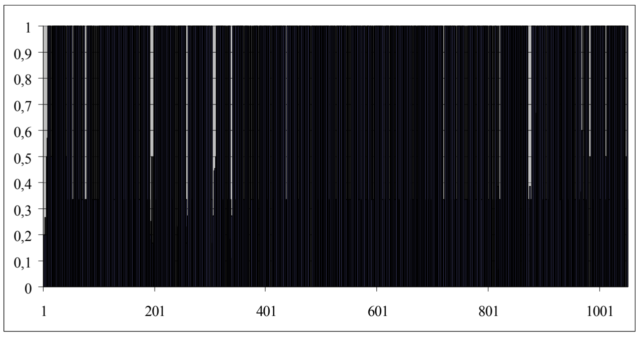


**Figure 14 Good reception rate for node 2 with CORIAS.**

In this medium density scenario, CORIAS demonstrates its efficiency regarding good reception rates compared to RANDOM.

## 7. CONCLUSION

In this paper a new strategy called CORIAS, to control collisions in random access slots in TDMA ad hoc networks, has been detailed. This algorithm, derived from Aloha slotted techniques used in RFID networks, tunes the number of nodes intending to access the channel in a same frame to guarantee a given probability of non collision. Due to the type of networks CORIAS has been developed for (tactical ad hoc networks) different mechanisms which allows to distribute the algorithm on all nodes have been developed. Simulation results have been provided to demonstrate the effectiveness of this new scheme. Although CORIAS has been designed for the transmission of HELLO messages, its underlying principles are much more general and may also be applied to spontaneous and uncommon transmissions when explicit time slot resource allocation is not efficient or possible. Moreover, a side effect of CORIAS when compared to a naïve random access scheme is that (1) the interference level generated on the random access slots is lower and (2) achieved performance is more predictable because of a lower variability in perceived collision probability. Future works include the assessment of the performance of CORIAS with a radio propagation model taking into account interferences, as well as possibility to switch between static and dynamic (contention based) access for signaling slots, as well as the definition of procedures to implement such transitions.

## 8. REFERENCES

[1] C. Perkins, E. Belding-Royer and S. Das ; "Ad hoc On-Demand Distance Vector (AODV) Routing" ; IETF RFC 3561; 2003

[2] T. Clausen and P. Jacquet ; "Optimized Link State Routing Protocol (OLSR)" ; IETF RFC 3626; 2003

[3] N. Abramson; "THE ALOHA SYSTEM – Another alternative for computer communications"; In Proceedings of Fall Joint Computer Conference AFIPS 1970

[4] "IEEE 802.11-2007: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications"; available at http://standards.ieee.org/getieee802/download/802.11-2007.pdf

[5] R.D. Soares and J.M.C. Brito; "Throughput comparison between RACH and CPCH in 3GPP"; In Proceedings of the Advanced Industrial Conference on Telecommunications/Service Assurance with Partial and Intermittent Resources Conference/Elearning on Telecommunications Workshop; 2005

[6] Jung-Gon Kim; "Adaptive slotted ALOHA-based anti-collision technique for RFID-Based Sensor Networks"; Proceedings of Wireless World Research Forum Conference, 2006.

[7] Xinqing Yan and Zhouping Yin and Youlun Xiong; "A Query Tree Dynamic Frame Slot ALOHA Collision Resolution Protocol for RFID Tags"; Future Generation Communication and Networking, 2008, Vol.1, p.198-201, IEEE Computer Society.