METHOD OF CONTENDING FOR ACCESS TO A COMMUNICATION CHANNEL AND ASSOCIATED COMMUNICATION DEVICE

Fig. 6

Abstract: The invention relates to contention for access to a communication medium. A method according to the invention includes, at a first node, steps of generating backoff values of a Collision Avoidance mechanism for a plurality of collaborative nodes, each backoff value being associated with a respective collaborative node; loading at least one hardware backoff counter with the backoff values, each hardware backoff counter being decremented over time until expiry; and upon detecting the expiry of one of the hardware backoff counters, accessing the medium if the backoff value loaded in the hardware backoff counter that has just expired is associated with the first node; otherwise, directly loading the hardware backoff counter that has just expired with the backoff value associated with a collaborative node different from the one with which the backoff value last loaded in that hardware backoff counter is associated.
METHOD OF CONTENDING FOR ACCESS TO A COMMUNICATION CHANNEL AND ASSOCIATED COMMUNICATION DEVICE

FIELD OF THE INVENTION

The present invention relates generally to communication networks and more specifically to methods and devices for contending for access to a communication channel.

BACKGROUND OF THE INVENTION

Wireless local area networks (WLANs), such as a wireless medium in a communication network using Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), are founded on the principles of collision avoidance. Such networks may also conform to a communication standard such as a communication protocol of 802.11 type e.g. Medium Access Control (MAC).

The IEEE 802.11 MAC standard defines the way WLANs must work at the physical and medium access control (MAC) level. Typically, the 802.11 MAC (Medium Access Control) relies on a contention-based mechanism based on the so-called “Carrier Sense Multiple Access with Collision Avoidance” (CSMA/CA) technique.

The standard 802.11 medium access protocol is mainly directed to the waiting management of communication nodes waiting for the medium to becomes idle so as to try to access to the medium, i.e. to the contention for access to the medium.

**Figure 1** illustrates a communication system in which several communication nodes exchange data packets over a radio transmission channel 100 of a wireless local area network (WLAN).

Access to the shared radio medium to send data packets is based on the CSMA/CA technique, for sensing the carrier and avoiding collision by separating concurrent transmissions in space and time.

Carrier sensing in CSMA/CA is performed by both physical and virtual mechanisms. The virtual carrier sensing is achieved by transmitting control packets to reserve the medium prior to transmission of data packets.

Then after, through the physical mechanism, a transmitting node first attempts to sense a medium that has been idle for at least one DIFS (standing for Distributed InterFrame Spacing) time period, before transmitting data packets.

However, if it is sensed that the shared radio medium is busy during the DIFS period, the transmitting node continues to wait until the radio medium becomes...
idle. To do so, it selects a backoff time (or value) from a uniformly distributed random variable with an interval \([0, CW]\), where \(CW\) (integer) is referred to as the contention window in the IEEE 802.11 standards, places the selected backoff value in a backoff counter and then starts a countdown of the backoff counter that decrements the backoff value by one each timeslot the medium remain idle for one network cycle. This backoff mechanism or procedure is the basis of the collision avoidance mechanism that defers the transmission time for a random interval, thus reducing the probability of collisions on the shared channel. After the backoff time period, the transmitting node may send the data packets.

One problem of wireless data communications is that it is not possible for the transmitting node to listen while sending, thus preventing the transmitting node from detecting data corruption due to channel fading or interference or collision phenomena. A transmitting node remains unaware of the corruption of the data packets sent and continues to transmit the packets unnecessarily, thus wasting access time.

The Collision Avoidance mechanism of CSMA/CA thus provides positive acknowledgement (ACK) of the sent data packets by the receiving node in case of successful packet reception, to notify the transmitting node that no corruption of the sent data packets occurred.

The ACK is transmitted at the end of reception of the data packet, immediately after a period of time called Short InterFrame Space (SIFS).

If the transmitting node does not receive the ACK within a specified ACK timeout or detects the transmission of a different packet on the channel, it may infer data packet loss. In that case, it generally reschedules the packet transmission according to the above-mentioned backoff procedure.

To improve the Collision Avoidance efficiency of CSMA/CA, a four-way handshaking mechanism is optionally implemented. One implementation is known as the RTS/CTS exchange, defined in the 802.11 standard.

The RTS/CTS exchange consists in exchanging control packets to reserve the radio medium prior to transmitting data packets during a transmission opportunity called TXOP in the 802.11 standard as described below, thus protecting data transmissions from any further collisions.

The backoff procedure is one of the first processes to be implemented in the communication nodes.
Figure 2 illustrates the behaviour of three groups of nodes during the back-off procedure: transmitting or source node 20, receiving or addressee or destination node 21 and other nodes 22.

Upon starting the backoff process 270 prior to transmitting data, a station e.g. transmitting node 20, initializes its backoff time counter to a random (backoff) value as explained above. The backoff time counter is decremented once every time slot interval 260 for so long as the radio medium is sensed idle (count down is starting from T0, 23 as shown in the Figure).

The time unit in the 802.11 standard is the slot time called ‘aSlotTime’ parameter. This parameter is specified by the PHY (physical) layer (for example, aSlotTime is equal to 9µs for the 802.11n standard). All dedicated space durations (e.g. backoff) are multiples of this time unit.

The communication nodes forming the same 802.11 cell of the radio network have quasi-synchronous clocks to ensure a consistent number of time slots 260 to be expired simultaneously by all communication nodes, e.g. during a backoff countdown procedure.

The backoff time counter is ‘frozen’ or suspended when a transmission is detected on the radio medium channel (countdown is stopped at T1, 24 for other nodes 22 having their backoff time counter decremented).

The countdown of the backoff time counter is resumed or reactivated when the radio medium is sensed idle anew, after a DIFS time period. This is the case for the other nodes at T2, 25 as soon as the transmission opportunity TXOP granted to transmitting node 20 ends and the DIFS period 28 lapses.

When the backoff time counter reaches zero 26 at T1, the timer expires, the corresponding node 20 is granted a TXOP, and the backoff time counter is reinitialized 29 using a new random backoff value.

In the example of the Figure implementing the RTS/CTS scheme, at T1, the transmitting node 20 that wants to transmit data packets 230 sends a special short frame or message acting as a medium access request to reserve the radio medium, instead of the data packets themselves, just after the channel has been sensed idle for a DIFS or after the backoff period as explained above.

The medium access request is known as a Request-To-Send (RTS) message or frame. The RTS frame generally includes the address of the receiving node ("destination 21") and the duration for which the radio medium is to be reserved for transmitting the control packets (RTS/CTS) and the data packets 330.
Upon receiving the RTS frame and if sensing the radio medium as being idle, the receiving node 21 responds, after an SIFS time period 27 (for example, SIFS is equal to 16 μs for the 802.11n standard), with a medium access response, known as a Clear-To-Send (CTS) frame. The CTS frame indicates the remaining time required for transmitting the data packets, computed from the time point at which the CTS frame starts to be sent.

The CTS frame is considered by the transmitting node 20 as an acknowledgment of its request to reserve the shared radio medium for a given time duration.

Thus, the transmitting node 20 expects to receive a CTS frame 220 from the destination node 21 before sending data 230 using unique and unicast (one source address and one addressee or destination address) packets.

The transmitting node 20 is thus allowed to send the data packets 230 upon correctly receiving the CTS frame 220 and after a new SIFS time period 27.

An ACK frame 240 is sent by the receiving node 21 after having correctly received the data packets sent, after a new SIFS time period 27.

If the transmitting node 20 does not receive the ACK 240 within a specified ACK Timeout, or if it detects the transmission of a different packet on the radio medium, it reschedules the packet transmission according to a new backoff procedure.

Since the RTS/CTS four-way handshaking mechanism 210/220 is optional in the 802.11 standard, it is possible for the transmitting node 20 to send data packets 230 immediately upon having its backoff time counter reach zero (i.e. at T1).

The requested time duration for transmission defined in the RTS and CTS frames defines the length of the granted transmission opportunity TXOP, and can be read by any listening node ("other nodes 22" in Figure 2) in the radio network.

To do so, each node has in memory a data structure known as the network allocation vector or NAV to store the time duration for which it is known that the medium will be busy. When listening to a control packet (RTS 210 or CTS 220) not addressed to itself, a listening node 22 updates its NAVs (NAV 255 associated with RTS and NAV 250 associated with CTS) with the requested transmission time duration specified in the control packet. The listening node 22 thus keeps in memory the time duration for which the radio medium will remain busy.

Access to the radio medium for the other nodes 22 is consequently deferred 30 by suspending 31 their associated timer and then by later resuming 32 the timer when the NAV has expired.
This prevents the listening nodes 22 from transmitting any data or control packets during that period.

It is possible that the destination node 21 does not receive the RTS frame 210 correctly due to a message/frame collision or to fading. Even if it does receive it, the destination node 21 may not always respond with a CTS 220 because, for example, its NAV is set (i.e. another node has already reserved the medium). In any case, the transmitting node 20 enters into a new backoff procedure.

The RTS/CTS four-way handshaking mechanism is very efficient in terms of system performance, in particular with regard to large packets since it reduces the length of the messages involved in the contention process.

In detail, assuming perfect channel sensing by each communication node, collision may only occur when two (or more) packets are transmitted within the same time slot after a DIFS 28 (DCF Inter-Frame Space) or when their own back-off counter has reached zero nearly at the same time (T1). If both transmitting nodes use the RTS/CTS mechanism, this collision can only occur for the RTS frames. Fortunately, such collision is early detected by the transmitting nodes since it is quickly determined that no CTS response is received.

As described above, collisions limit the optimal functioning of the radio network, and the random backoff procedure is used as the fundamental approach for supporting distributed access to the medium among communication nodes, in the IEEE 802.11 standards, in particular in the emerging IEEE 802.11n/ac standards.

A major drawback of the random backoff procedure lies in that the randomly chosen backoff value for the backoff counter may degrade the utility of the medium and thus degrade the performance of carrier sense multiple access (CSMA) technique, especially when a large number of communication nodes are competing (i.e. in saturated traffic scenarios).

To cope with this problem, some mechanisms have been developed to provide a collaborative access to the network medium. These mechanisms aim at providing a “low-latency/low-collision” communication service on top of classical 802.11 CSMA/CA.

An example is given in publication GB 2,490,963 where a group of communication nodes, referred to as collaborative communication nodes (the other nodes are “legacy” nodes), organize together to provide a “backoff synchronization”: access collaboration is performed by controlling the “random backoff values” used to
initialize the backoff time counters among the group of collaborative nodes in order that no backoff count duplication occurs.

To do so, each node of the collaborative group computes, in a deterministic fashion, the backoff value each other collaborative node will use and monitors the decrementing of these backoff values. Due to the deterministic approach, the collaborative nodes obtain the same backoff values for the same nodes.

The collaborative nodes thus maintain the same list of backoff values associated with the community, including its local backoff value, and decrement corresponding backoff counters over time. This makes it possible for it not to re-use a backoff value currently existing for any of the other collaborative nodes when generating a new local backoff value. Thus, collisions are eliminated (or greatly reduced) inside the community of collaborative communication nodes.

The effective use of the deterministic backoff approach relies on the ability of each collaborative communication node to "hear" transmissions from the other communication nodes. This is necessary to synchronize or resynchronize their backoff countdowns.

In contrast to theory (or software simulation environment on which GB 2,490,963 relies), 802.11 commodity hardware suffers from limitations to support such a backoff-synchronized access scheme. Indeed, the hardware products usually possess a limited number of backoff counter resources (typically one to four backoff registers in case of QoS support) and a limited number of random generator resources (usually a single one) to generate the backoff values.

Those hardware limitations prevent efficient handling of deterministic backoff values for all the collaborative nodes within the same collaborative node considered. This is because it is ideally expected there to be one generator and one backoff counter per collaborative node to be monitored, whereas the hardware does not provide so many resources. Consequently, the implementation of such a backoff-synchronized access scheme is difficult in hardware having limited resources.

The present invention has been devised to address at least one of the foregoing concerns.

**SUMMARY OF THE INVENTION**

According to first embodiments of the invention, there is provided a method of contending for access to a communication medium shared between communication nodes, comprising at a first communication node:
generating backoff values driving backoff procedures of a Collision
Avoidance mechanism for a plurality of collaborative communication nodes including
the first communication node, each backoff value being associated with a respective
collaborative communication node;

loading at least one hardware backoff counter with the backoff values, each
hardware backoff counter being decremented over time until expiry before loading a
new backoff value therein; and

upon detecting the expiry of one of the hardware backoff counters,
accessing the communication medium if the backoff value loaded in the
hardware backoff counter that has just expired is associated with the first
communication node, and loading a new backoff value in the hardware backoff counter
that has just expired,

otherwise, directly loading the hardware backoff counter that has just
expired with the backoff value associated with a collaborative communication node
different from the one with which the backoff value last loaded in that hardware backoff
counter is associated.

The present invention makes it possible to have collaborative management
of the medium access in a low-resource environment with a number of hardware
backoff counters less than the number of collaborative nodes, while still providing a
contention-based access to any of the nodes.

This is achieved thanks to the successive loading of backoff values
associated with different collaborative nodes in the same hardware backoff counter. As
a result, the first communication node keeps monitoring the medium accesses of all the
other collaborative nodes, thus allowing synchronization to be performed for example.

As the number of backoff values that can be successively loaded in the
same hardware backoff counter is not limited, the present invention is fully adapted to a
varying number of collaborative nodes (i.e. to a varying number of backoff values
resulting from the entering or the exit of one or more nodes from the community).

Correspondingly, there is provided a communication device contending for
access to a communication medium shared between communication nodes, the
communication device comprising:

a backoff generator configured for generating backoff values driving backoff
procedures of a Collision Avoidance mechanism for a plurality of collaborative
communication nodes including the communication device, each backoff value being
associated with a respective collaborative communication node;
a backoff loader configured for loading at least one hardware backoff counter with the backoff values, each hardware backoff counter being decremented over time until expiry before loading a new backoff value therein; and the communication device being configured, upon detecting the expiry of one of the hardware backoff counters, for:

accessing the communication medium if the backoff value loaded in the hardware backoff counter that has just expired is associated with the communication device, and loading a new backoff value in the hardware backoff counter that has just expired,

otherwise, directly loading the hardware backoff counter that has just expired with the backoff value associated with a collaborative communication node different from the one with which the backoff value last loaded in that hardware backoff counter is associated.

The communication device has the same advantages as the method defined above.

Further embodiments of the invention are defined in the dependent appended claims and are explained below in terms of method features.

In embodiments, the generated backoff values are successively loaded in one and the same hardware backoff counter of the first communication node, and the method comprises, upon detecting the expiry of the hardware backoff counter, determining whether or not the backoff value last loaded in the hardware backoff counter that has just expired is associated with the first communication node.

In these embodiments, the same backoff counter thus receives backoff values of different natures: sometimes to trigger (when expiring) an access to the communication medium for the first communication node having the hardware backoff counter, and other times, only to provide a monitoring of the other collaborative nodes (for example with a view of keeping synchronization as described below) without medium access for the first communication node. The same hardware backoff counter has thus a double function.

In other embodiments, the generated backoff value associated with the first communication node is loaded in a first hardware backoff counter of the first communication node to be decremented, and the other generated backoff values are successively loaded in at least one other hardware backoff counter of the first communication node. In practice, a hardware implementation of a communication node
includes at most four hardware backoff counters, meaning that the other hardware backoff counters are generally one, two or three.

These other embodiments keep the conventional handling of the “local” backoff procedure (i.e. for the first communication device having the backoff counters) while providing an efficient management and monitoring of the backoff procedures of the other collaborative nodes, at low resource cost.

Thus, this is particularly efficient in a low-resource environment such as when the first communication node includes less backoff counters than the number of collaborative communication nodes.

In yet other embodiments, upon detecting the expiry of one hardware backoff counter previously loaded with the backoff value associated with a given collaborative communication node, a new backoff value is generated for said given collaborative communication node. This allows the given node to be granted a medium access again after a new backoff procedure.

According to specific features, the hardware backoff counter that has just expired is loaded with the smallest backoff value from amongst the newly generated backoff value and the backoff values associated with the other collaborative communication nodes. This is to give priority to the monitoring of the backoff procedure that will end first. This means that an efficient synchronization of the first node with the community can be achieved with low processing cost. In addition, this approach is compliant with the conventional backoff procedure for each collaborative node.

According to specific embodiments, the smallest generated backoff value so loaded is subtracted from each remaining backoff value not yet loaded, and the resulting backoff values are stored in a memory of the first communication node. This is for the first node to keep track of and still efficiently monitor the backoff procedures of the other collaborative nodes, despite the insufficient number of backoff counters. This is because the resulting backoff values so stored already take the values they should have when the backoff counter expires. Upon such expiry, the backoff values in memory are thus in line with the backoff procedure of each of the collaborative nodes.

As a consequence, less computation is required when loading a new backoff value in the backoff counter after the expiry of the latter.

In yet other embodiments, the step of generating the backoff values comprises generating only one backoff value each time the hardware backoff counter
expires, and the newly generated backoff value is associated with one of the collaborative communication nodes according to a predefined node order (e.g. Round-robin scheduling) of the collaborative communication nodes and is loaded in the hardware backoff counter to be decremented. This configuration results in having the first communication node comprising only one backoff value at the same time, in case only one hardware backoff counter is available.

Thus, these embodiments reduce memory requirements in the first communication node compared to the embodiments where a backoff value is handled for each collaborative node and needs to be stored in memory.

In other embodiments where two or more hardware backoff counters are available, the step of generating the backoff values comprises generating only one backoff value each time a hardware backoff counter expires, and if the hardware backoff counter that has just expired is the first hardware backoff counter, the newly generated backoff value is associated with the first communication node and is loaded in the first hardware backoff counter, otherwise, the newly generated backoff value is associated with one of the other collaborative communication nodes according to a predefined node order and is loaded in the hardware backoff counter that has just expired.

This configuration saves memory when handling the backoff values of the other collaborative nodes. Again, this configuration results in having the first communication node comprising, at any time, as many backoff values as the number of hardware backoff counters, but not more.

According to specific features, non-collaborative communication nodes (i.e. “legacy” nodes) access the communication medium based on a contention window, and the backoff values for the collaborative nodes are generated using a uniformly distributed random variable with an interval [0, CCW], where CCW is the size of a collaborative contention window strictly less than the size of the contention window.

One reason for implementing such an approach is that handling only one backoff value at the same time for all or part of the collaborative nodes results in adding the backoff values before a given collaborative node is granted access to the communication medium. To be more precise, the given collaborative node has to wait for the successive expiries of the backoff values associated with all the collaborative nodes that are prior to it according to the predefined node order. This may take a long time. The shorter collaborative contention window thus seeks to decrease the impact of such cumulative wait, by statistically reducing individually each backoff value.
According to another feature, CCW is a function of the size of the contention window and of the number of collaborative communication nodes. This is to adapt the decreasing of the impact defined above, because the more the number of collaborative nodes, the longer the nodes wait to be granted an access to the communication medium.

In yet other embodiments, the method further comprises, at the first communication node, performing a synchronization adjustment process of the one or more hardware backoff counters upon receiving a message from another collaborative communication node at a time drifted from the expiry of the backoff value associated with said other collaborative communication node. This is because a main purpose of the monitoring of the other collaborative nodes is to maintain synchronization in the community of collaborative nodes.

In practice, the received message has generally been sent by the collaborative node the backoff value of which is currently being decremented in the hardware backoff counter that next expires or the backoff value of which is the last expired. Otherwise, this would mean that another backoff value should have expired first.

However other situations may occur, in particular in case the synchronization has been lost for a long time or a collaborative node has an erratic behaviour.

According to specific features, the synchronization adjustment process comprises subtracting a time drift from the one or more hardware backoff counters when the message is received before the expiry of the hardware backoff counter decrementing the backoff value associated with said other collaborative communication node, the time drift being equal to the backoff value associated with said other collaborative communication node that is being decremented when receiving the message.

This means that the hardware backoff counter decrementing the backoff value associated with the message sender is set to 0, while the other counters, if any, are decreased by the same amount (as explained above, unless there is erratic behaviour, the other counters are higher than the one set to 0, since the message is expected to be received from the collaborative node which has the next access to the communication medium).
The above provision thus makes it possible to obtain an efficient re-synchronization of the first communication node with the community, in case it lags behind the other relative to the network clock.

Other specific features may regard the case where the first communication node is ahead of the other collaborative nodes, rather than lagging behind. If the message is received with a time drift after the expiry of the hardware backoff counter decrementing the backoff value associated with the other collaborative communication node having sent the message and a new backoff value has been loaded in the same hardware backoff counter after its expiry, the synchronization adjustment process comprises, upon receiving the message, resetting the hardware backoff counter with the new backoff value as previously loaded.

In addition, the method may further comprise modifying any other hardware backoff counter by the same amount of time, i.e. by the time drift (e.g. said new backoff value less its value when receiving the message).

Of course, if other hardware backoff counters have expired in the meantime (during the time drift), they should be restored as they were when the message sender’s backoff counter expired. This may thus require saving in memory an image (snapshot) of the hardware backoff counters and of any backoff values in memory upon such expiry, to be able to restore these values and counters.

Again the above provision makes it possible to re-synchronize the first communication node with the community.

The invention is also directed to a non-transitory computer-readable medium storing a program which, when executed by a microprocessor or computer system in a communication device sharing a communication medium with other communication nodes, causes the communication device to perform contention for access to the communication medium, including the steps of:

- generating backoff values driving backoff procedures of a Collision Avoidance mechanism for a plurality of collaborative communication nodes including the first communication node, each backoff value being associated with a respective collaborative communication node;
- loading at least one hardware backoff counter with the backoff values, each hardware backoff counter being decremented over time until expiry before loading a new backoff value therein; and
- upon detecting the expiry of one of the hardware backoff counters,
accessing the communication medium if the backoff value loaded in the hardware backoff counter that has just expired is associated with the first communication node, and loading a new backoff value in that hardware backoff counter that has just expired,

otherwise, directly loading the hardware backoff counter that has just expired with the backoff value associated with a collaborative communication node different from the one with which the backoff value last loaded in that hardware backoff counter is associated.

The non-transitory computer-readable medium may have features and advantages that are analogous to those set out above and below in relation to the method and device, in particular that of providing a collaborative management of the medium access in low-resource environment.

Another aspect of the invention relates to a method of contending for access to a communication medium shared between communication nodes substantially as herein described with reference to, and as shown in, Figure 6; Figure 9; Figure 10; Figure 6, 7 and 8; Figure 7, 8 and 10 of the accompanying drawings.

Yet another aspect of the invention relates to a communication device contending for access to a communication medium shared between communication nodes substantially as herein described with reference to, and as shown in, Figure 5 of the accompanying drawings.

At least parts of the method according to the invention may be computer implemented. Accordingly, the present invention may take the form of an entirely hardware embodiment, an entirely software embodiment (including firmware, resident software, micro-code, etc.) or an embodiment combining software and hardware aspects which may all generally be referred to herein as a "circuit", "module" or "system". Furthermore, the present invention may take the form of a computer program product embodied in any tangible medium of expression having computer usable program code embodied in the medium.

Since the present invention can be implemented in software, the present invention can be embodied as computer readable code for provision to a programmable apparatus on any suitable carrier medium, for example a tangible carrier medium or a transient carrier medium. A tangible carrier medium may comprise a storage medium such as a floppy disk, a CD-ROM, a hard disk drive, a magnetic tape device or a solid state memory device and the like. A transient carrier medium may include a signal such as an electrical signal, an electronic signal, an optical signal, an
acoustic signal, a magnetic signal or an electromagnetic signal, e.g. a microwave or RF signal.

**BRIEF DESCRIPTION OF THE DRAWINGS**

5 Embodiments of the invention will now be described, by way of example only, and with reference to the following drawings in which:

**Figure 1** illustrates a typical wireless communication system in which embodiments of the invention may be implemented;

**Figure 2** is a timeline schematically illustrating the conventional communication mechanism according to the IEEE 802.11 standard;

**Figure 3** is a timeline schematically illustrating a backoff-based collaborative medium access scheme shared by a group of collaborative nodes;

**Figure 4** is a block diagram illustrating components of a communication device in which embodiments of the invention may be implemented;

**Figure 5** illustrates function blocks of the same communication device;

**Figure 6** illustrates, using a flowchart, general steps of a first implementation of the invention;

**Figure 7** illustrates an exemplary implementation of the backoff list generation of **Figure 6**;

**Figure 8** illustrates the evolving of the backoff list as successive medium accesses are granted to the community of collaborative nodes, according to the first implementation of **Figure 6**;

**Figure 9** illustrates, using a flowchart, general steps of a second implementation of the invention; and
Figure 10 illustrates, using a flowchart, general steps of a third implementation of the invention.

**DETAILED DESCRIPTION OF EMBODIMENTS OF THE INVENTION**

The invention provides methods, devices and systems for contending for access to a communication medium of an ad-hoc wireless network, which communication medium being shared between a plurality of communication nodes or devices.

As introduced above, a goal of the access scheme according to the invention in radio networks is to allow collaborative synchronization of the nodes similarly to GB 2,490,963, when the nodes have low hardware resources, in particular few hardware backoff counters to monitor the backoff counters of the collaborative nodes. In GB 2,490,963, each collaborative node generates backoff values driving backoff procedures of a Collision Avoidance mechanism for itself and for the other collaborative communication nodes, each backoff value being associated with a respective collaborative communication node; and loads at least one hardware backoff counter with the backoff values, each hardware backoff counter being decremented over time until expiry before loading a new backoff value therein.

A communication device according to the invention achieves this above-defined goal by successively loading backoff values corresponding to different collaborative nodes into the same hardware backoff counters. To allow such mechanism while still performing an appropriate backoff procedure for itself, the communication device, upon detecting the expiry of one of its hardware backoff counters, accesses the communication medium if the backoff value loaded in the hardware backoff counter that has just expired is associated with itself, and loads a new backoff value in the hardware backoff counter that has just expired; otherwise, directly loads the hardware backoff counter that has just expired with the backoff value associated with a collaborative communication node different from the one with which the backoff value last loaded in that hardware backoff counter is associated.

The communication device thus mixes its own backoff procedure with the monitoring of the backoff value countdown for the other collaborative nodes, using few hardware backoff counters.

The following description focuses on a radio network having collaborative nodes that share a collaborative radio medium access scheme to access the radio medium, thus providing free collision within the collaborative group as soon as they can
each monitor the backoff values of the other collaborative nodes. The RTS/CTS mechanism involved in the examples below remains optional as in the IEEE 802.11 standard.

In the example of a 802.11-based wireless communication network implementing CSMA/CA mechanism with RTS/CTS exchange as shown in Figures 1 and 2, several communication nodes 101, 102, 103, 104, 105, 106, 107 exchange data packets through a transmission channel 100 that may drop or corrupt the data packets.

The communication nodes can be divided into two groups: a first group of $Nbi$ nodes 101 to 103 (where i is an integer between 1 and 3) implementing the conventional 802.11 standard but not the invention (they are named legacy nodes since each of them is independent and does not interact or cooperate with each other), and a second group of $Ncj$ nodes 104 to 107 (where j is an integer between 1 and 4) implementing the 802.11 standard and the mechanisms of the invention to control radio medium access.

For the sake of illustration, all data exchanges between nodes of the communication system, that is to say nodes $Nc_i$ and nodes $Nb_i$, are managed through the standardized 802.11n MAC/PHY layers.

Collaborative radio medium access consists in a distributed mechanism providing a distinct backoff count for each collaborative node of the group, in order that no backoff time count duplication occurs. It results that collision avoidance in medium access is provided within the collaborative group.

An additional feature of the collaborative group is the sharing of granted transmission slot: once the backoff count reduces to zero for one collaborative node, that collaborative node reserves medium access (through classical RTS/CTS scheme) for the whole group and may let the group share this granted 802.11 timeslot between all or part of the collaborative nodes (those which have data to be sent, based on a TDMA mechanism). As a result, this collaborative sub-network allows low jitter and regular access onto the radio medium for the collaborative nodes, which is extremely useful for conveying data of real-time acquisition applications.

Figure 3 schematically illustrates the mechanisms within the collaborative group to provide a free collision collaborative radio medium access scheme. An example of such scheme is described in publication GB 2,490,963.

To ensure free collision within the collaborative group, the collaborative nodes use different backoff values, the expiries of which thus occurring at different times. More precisely, collaborative radio medium access is performed by watching the
“random values” used to initialize the backoff time counters among the collaborative group, in order that no backoff time count duplication occurs.

As an example, the collaborative node may implement the same pseudo-random generator algorithm using node IDs to generate the backoff value. In this situation, each collaborative node is able to know the current backoff value of each collaborative node. It will use the pseudo-random generator algorithm once or several times until generating an own backoff value that is different from the current backoff values (being decremented according to the conventional backoff procedure). Thus no duplication of the backoff values occurs for the collaborative nodes, meaning that each node of the collaborative group tries to access the communication medium in a back-off slot distinct from the other collaborative nodes. This mechanism thus avoids any access collision among the collaborative nodes.

As a variant, each collaborative node may transmit periodically its backoff value to the other collaborative nodes.

Each collaborative node thus has a virtual local image storing the current backoff time counter values of the other collaborative nodes.

In Figure 3, the backoff slots 260 have been numbered. The idea of the collaborative scheme is to position the expiry of the backoff counters of the collaborative nodes to different backoff slots or values: as example, by considering an absolute scale of backoff slots, the current backoff slot being number ‘n’ 300, the collaborative scheme 330 would select ‘n+3’ 310 and ‘n+10’ 320 for two distinct collaborative nodes.

Each collaborative node can follow the decrementing of the backoff time values of the other collaborative nodes and can predict any attempt to access the wireless medium by such other nodes. Since it has a different backoff time value, the collaborative node will access the network in a distinct backoff slot to the other nodes, avoiding any access collision among the group of collaborative nodes.

In the example of Figure 3, the collaborative node having its backoff counter expiring at backoff slot ‘n+3’ sends data during the transmission slot 230. Of course the RTS/CTS mechanism may be implemented prior to the transmission slot 230.

In a variant, the transmission slot 230 may be shared with all or part of the collaborative nodes. For example it may be decided to split 230 into a plurality of elementary transmission time slots, each being allocated to one collaborative node, using TDMA. The order for allocation may be the order of the node IDs known by each collaborative node.
The knowledge of backoff slot positioning information allows the collaborative nodes to synchronize their accesses to the communication medium and avoid collisions. However, various concurrent communications according to EDCA behavior (each EDCA queue has its own backoff engine) may introduce a loss of synchronization of the backoff counters which are setup among the group, leading to medium collision inside the group. As described below, resynchronization is contemplated in embodiments of the invention.

Figure 4 schematically illustrates a communication device 400 of the radio network 100, configured to implement at least one embodiment of the present invention when it is one of the nodes $N_{Ci}$. The communication device 400 may be a device such as a micro-computer, a workstation or a light portable device. The communication device 400 comprises a communication bus 413 to which there are preferably connected:

- a central processing unit 411, such as a microprocessor, denoted CPU;
- a read only memory 407, denoted ROM, for storing computer programs for implementing the invention;
- a random access memory 412, denoted RAM, for storing the executable code of methods according to embodiments of the invention as well as the registers adapted to record variables and parameters necessary for implementing methods according to embodiments of the invention; and
- at least one communication interface 402 connected to the radio communication network 100 over which digital data packets are transmitted, for example a wireless communication network according to the 802.11n protocol. The data packets are written from a FIFO sending memory in RAM 412 to the network interface for transmission or are read from the network interface for reception and write into a FIFO receiving memory in RAM 412 under the control of a software application running in the CPU 411.

Optionally, the communication device 400 may also include the following components:

- a data storage means 404 such as a hard disk, for storing computer programs for implementing methods according to one or more embodiments of the invention;
- a disk drive 405 for a disk 406, the disk drive being adapted to read data from the disk 406 or to write data onto said disk;
a screen 409 for displaying decoded data and/or serving as a graphical interface with the user, by means of a keyboard 410 or any other pointing means.

The communication device 400 can be connected to various peripherals, such as for example a digital camera 408, each being connected to an input/output card (not shown) so as to supply data to the communication device 400.

The communication bus provides communication and interoperability between the various elements included in the communication device 400 or connected to it. The representation of the bus is not limiting and in particular the central processing unit is operable to communicate instructions to any element of the communication device 400 directly or by means of another element of the communication device 400.

The disk 406 can be replaced by any information medium such as for example a compact disk (CD-ROM), rewritable or not, a ZIP disk or a memory card and, in general terms, by an information storage means that can be read by a microcomputer or by a microprocessor, integrated or not into the apparatus, possibly removable and adapted to store one or more programs whose execution enables a method according to the invention to be implemented.

The executable code may be stored either in read only memory 407, on the hard disk 404 or on a removable digital medium such as for example a disk 406 as described previously. According to a variant, the executable code of the programs can be received by means of the communication network 403, via the interface 402, in order to be stored in one of the storage means of the communication device 400, such as the hard disk 404, before being executed.

The central processing unit 411 is adapted to control and direct the execution of the instructions or portions of software code of the program or programs according to the invention, which instructions are stored in one of the aforementioned storage means. On powering up, the program or programs that are stored in a non-volatile memory, for example on the hard disk 404 or in the read only memory 407, are transferred into the random access memory 412, which then contains the executable code of the program or programs, as well as registers for storing the variables and parameters necessary for implementing the invention.

In this embodiment, the apparatus is a programmable apparatus which uses software to implement the invention. However, alternatively, the present invention may be implemented in hardware (for example, in the form of an Application Specific Integrated Circuit or ASIC).
Figure 5 is a block diagram schematically illustrating the architecture of a node 400 adapted to carry out, at least partially, the invention, for example a node Nc of Figure 1. As illustrated, node 400 comprises a physical (PHY) layer block 503, a MAC layer block 502, and an application layer block 501.

The PHY layer block 503 (here a 802.11 standardized PHY layer) has the task of formatting and sending or receiving data packets over the radio medium used 100, such as a medium access request of the RTS type to reserve a transmission slot, a medium access response of the CTS type to acknowledge reservation of a transmission slot, as well as of data packets to/from that radio medium.

The MAC layer block 502 comprises a standard MAC 802.11 layer 504 and additional blocks 505 to 508 for carrying out, at least partially, the invention. The MAC layer block 502 may be implemented in software, which software is loaded into RAM 312 and executed by CPU 311.

As an example, the node data transmission block 505 performs the task of exchanging data packets with the PHY layer 503 and the application layer 501, in particular in the process of accessing the communication medium.

The backoff management block 506 has the task of managing the backoff values used by the node in a collision avoidance scheme according to the invention. That is, it manages its own backoff value and the backoff values of the other collaborative nodes according to the collaborative backoff procedure as introduced above. There are situations where the backoff management block 506 manages a number of hardware backoff counters that is less than the number of collaborative nodes currently operating. The present invention has particularly interest in such situations.

The node data transmission block 505 works together with the backoff management block 506 to, upon detecting the expiry of one of the hardware backoff counters, access the communication medium if the backoff value loaded in the hardware backoff counter that has just expired is associated with the current node, and loads a new backoff value in the hardware backoff counter that has just expired; otherwise, directly loads the hardware backoff counter that has just expired with the backoff value associated with a collaborative communication node different from the one with which the backoff value last loaded in that hardware backoff counter is associated.

Finally, the application layer block 501 implements an application that generates and receives data packets, for example data packets of a video stream.
Figures 6 to 8 illustrate a first implementation of the invention in which the collaborative node considered ("local" node) has only one hardware backoff counter to manage its own backoff value and the backoff values of the other collaborative nodes.

In this first exemplary implementation, the generated backoff values are successively loaded in one and the same hardware backoff counter of the node, and, upon detecting the expiry of the hardware backoff counter, it is determined whether or not the backoff value last loaded in the hardware backoff counter that has just expired is associated with the node. This is to decide whether this time, the communication medium has to be accessed or a new backoff value corresponding to another collaborative node has to be loaded in the counter.

Figure 6 illustrates, using a flowchart, general steps of the first implementation of the invention.

The process of the Figure is distributed and performed by every node belonging to the community, i.e. every collaborative node.

In this implementation, the system of the nodes manages the whole community synchronization with only one hardware backoff counter and one random generator per collaborative node. Despite the single hardware backoff counter, this implementation makes it possible to efficiently monitor the backoff values (i.e. virtual backoff counters) of all the collaborative nodes, while each collaborative node accesses the communication medium according to its own backoff procedure. The same result as if there were as many backoff counters decrementing backoff values as the number of collaborative nodes is achieved, with less hardware resources.

The process of Figure 6 is performed within each collaborative node. For the description below, one collaborative node is considered and called "local node".

The process starts at step 600 in which the local node obtains a seed shared by all the community. One of the collaborative nodes may have sent this seed to all group nodes.

Note that this seed sharing is essential in order to initialize the random generator embedded by each node to generate the backoff values for each collaborative node. Thus, the nodes of the community will be able to produce exactly the same random number series as backoff values and thus prevent collision.

Other alternatives to obtain the seed without message exchange are available to initialize the random generator. For instance, the use of a static number configured in factory; or the use of the node identifier (node_id) of one of the collaborative nodes as a seed.
A function such as a pseudo-random generator is similarly implemented in all the collaborative nodes, which function takes this shared seed as a function parameter. This makes it possible to provide the same random value computation algorithm to all collaborative nodes. Accordingly, the collaborative nodes are able to obtain the same random number series, meaning to determine a new backoff value associated with one of the nodes of the community, the backoff values determined in each node of the community being the same.

Steps 605-620 ensure a distinct backoff value to be allocated to each node forming the backoff-synchronized community.

Once the random generator is initialized with the commonly-known seed, the system generates a backoff value at step 605. This first backoff value is assigned to a first node of the community (as an example, the nodes are sorted by rising node id).

Next, at step 610, the local node tests whether or not one backoff value is assigned to each community node. If it is not, the system loops back to step 605 to generate another backoff value for another remaining collaborative node.

These steps 605-610 makes it possible to generate the backoff values at the start of the node, but also to generate any new backoff value when the hardware backoff counter has entirely decremented a previous backoff value (through the loop from steps 655-660 to step 605 as described below).

If it is (i.e. each collaborative node is associated with one backoff value), the node checks whether or not duplicate backoff values have been assigned to different nodes. This is step 615. If duplicate backoff values exist in the backoff list, a new backoff value is drawn to replace one of the duplicate values to thus prevent collisions inside the community. When each node is associated to a unique backoff value, the system orders the backoff list (for instance in rising order of the backoff values). This is step 620.

Next, the hardware backoff counter is initialized with the minimum backoff value of the list, at step 625. This happens at the start of the node or each time the hardware backoff counter expires (through the loop from steps 655-660 to step 605 as described below). In other words, the hardware backoff counter that has just expired is loaded with the smallest backoff value from amongst the newly generated backoff value and the backoff values associated with the other collaborative communication nodes.
Next, an update of the other backoff values of the list is performed at step 630. Because these backoff values will not be decremented by a backoff counter, due to the lack of hardware backoff counters, step 630 provides to store these values with the future values they will have upon expiry of the hardware backoff value loaded at step 625.

To perform this update 630 of the backoff values, the smallest generated backoff value so loaded in the hardware backoff counter is subtracted from each remaining backoff value not yet loaded, and the resulting backoff values are stored in a memory of the node considered.

**Figure 7** illustrates an exemplary implementation of the backoff list generation. Three main steps are shown in the Figure, namely the backoff generation (table 700 corresponding steps 610-615), the backoff list ordering (table 730 corresponding to step 620) and the backoff list update (table 740 corresponding to step 630).

In this example, the random set generated by the random generator based on the seed obtained at step 600 is the following: 15, 3, 8, 11, 1, 7 and 12. Moreover, in this example, the community is made up of four collaborative nodes (numbered from 0 to 3).

Column 710 represents the collaborative nodes, where all the nodes are identified with their node identifier (node_id). In this first table 700, the nodes are arranged by ascending node_ids. Column 720 represents the backoff value associated with each collaborative node using the random set generated.

Table 700 represents the result of the execution of steps 605 and 610. One backoff value is assigned to each collaborative node, including the local node in which **Figure 6** is executed.

With a first execution of step 605, the system draws a value 15 for Node 0. In the backoff list, the backoff value 15 (721) is thus assigned to Node 0 (711).

With a second execution of step 605, value 3 (722) is generated for Node 1 (712). With a third execution, the backoff value 8 (723) is obtained for Node 2 (713). And with a last execution of step 605, the backoff value 11 (724) is obtained for Node 3 (714). As a result of this iterative backoff generation, there is no duplicate backoff value (no internal collision) among the nodes of the community. The process thus goes to step 620.

Table 730 is the result of ordering step 620 performed on the backoff values of the list. In this example, the backoff values are arranged in rising order.
As a consequence, Node 1 (712) having the smallest backoff value becomes the first in the backoff list, and Node 2 (713) becomes the second in the list and so on.

Once the backoff list ordering has been done, the process loads the minimum backoff value (722), i.e. backoff value 3 corresponding to Node 1, in the hardware backoff counter before going to updating step 630.

Table 740 is the result of the update process 630. The process subtracts the minimum backoff value (722), ‘3’ in the example, to all the backoff values. With such operation, the resulting backoff value list is ready for the next iteration of the backoff generation and especially for the duplication process (step 615) in which the new backoff value is compared to the current values of the backoff list.

Note that the first entry of the list may be deleted from the list since it has been loaded in the hardware backoff counter and is thus no longer important for the next iteration. This saves memory in the local node.

Back to Figure 6, after the update step 630, the conventional backoff countdown procedure is performed on the hardware backoff counter, as long as the communication medium is sensed idle, meaning that it is suspended when a transmission is detected on the communication medium.

The countdown is shown by step 635 in the Figure: as long as the communication medium is sensed idle, the local node checks whether or not the backoff time elapsed i.e. the hardware backoff counter expires by reaching 0.

When the communication medium becomes busy, the local node suspends the countdown and tests whether or not the data received over the communication medium come from one collaborative node of the community (test 640). For example, the data received are generally a RTS message when the four-way handshaking mechanism is implemented.

If the local node receives a RTS message from a collaborative node while the backoff counter has not yet expired, a de-synchronization among the collaborative nodes is thus detected. The process thus goes to step 655 where a re-synchronization operation is performed.

Otherwise (if the data received are from a legacy node – output “no” at step 640), the process loops back to step 635 to wait for the communication medium to be idle again in order to resume the backoff countdown procedure. Thus, the local node waits for backoff time expiration or for a new transmission from another node.
When the backoff counter expires, the process goes to step 645 to check whether or not the local node is the owner of the backoff value that has been decremented in the counter (i.e. whether or not the backoff value loaded in the hardware backoff counter that has just expired is associated with the local node).

This check may be based on the backoff list described above with reference to Figure 7. In this list, the local node stores each backoff values in association with its owner, i.e. a collaborative node. The backoff list may include as many entries as the number of collaborative node (if the backoff value loaded in the hardware backoff counter and thus set to 0 at step 630 is kept) or a number of entries equal to the number of collaborative nodes less 1 (the one the backoff value has been loaded in the counter).

Thus the check at step 645 may simply consist to test whether or not the first entry (the backoff value of which is the minimum one, set to 0) of the backoff list is associated with the local node (test on node_id), or whether or not the sole collaborative node that is missing in the backoff list is the local node.

If the backoff counter has expired for the local node (output “yes” at step 645), it means that the local node is granted a medium access for a TXOP. As a consequence, the local node attempts to access the communication medium by sending a RTS message at step 660. A conventional communication using a TXOP is thus performed as described above with respect to Figure 3 before looping back to step 605 to generate a new backoff value for the local node.

Otherwise (the backoff counter has expired for another collaborative node – output “no” at step 645), the local node waits for a RTS message to be received from said other collaborative node. This is step 650.

If the local node receives such a RTS message, the process loops back to step 605 to generate a new backoff value for the RTS sender.

If the local node does not receive such a RTS message or receives it with a delay or time drift, it goes to step 655 to perform a re-synchronization operation (synchronization adjustment step).

As it transpires from the above, the local node performs a synchronization adjustment process of the hardware backoff counter upon receiving a message from another collaborative communication node at a time drifted (or shifted) from the expiry of the backoff value associated with said other collaborative communication node.

An aim of the synchronization adjustment process is to resynchronize the backoff values of the list when the local node detects a mismatch between the
community behavior and its image (based on current backoff counter) of the community.

Two main situations occur: when the local node is lagged behind the other collaborative nodes (output "yes" at test 640), or when the local node is ahead of the other collaborative nodes (output "no" at test 650).

In the first situation (lagged behind), the synchronization adjustment process comprises subtracting a time drift from the hardware backoff counter when the (RTS) message is received before the expiry of the hardware backoff counter decrementing the backoff value associated with said other collaborative communication node, the time drift being equal to the backoff value associated with said other collaborative communication node that is being decremented when receiving the message. This is to temporally realign the local backoff counter with the expiry of the backoff procedure at the other collaborative node.

For instance, if the local node receives a RTS from another collaborative node at step 640 while the hardware backoff counter having the associated backoff value is different from 0, the local node subtracts the current value of the backoff counter from the same backoff counter value, to resynchronize it with the community.

In the second situation (temporally ahead), if the (RTS) message is received with a time drift after the expiry of the hardware backoff counter decrementing the backoff value associated with the other collaborative communication node having sent the message and a new backoff value has been loaded (possibly after a short waiting delay, e.g. equal to DIFS 28) in the same hardware backoff counter after its expiry, the synchronization adjustment process comprises, upon receiving the message, resetting the hardware backoff counter with the new backoff value as previously loaded.

In a slight variant, if no new backoff value has been loaded (for instance because the local node does not load another backoff value as long as the RTS message has not been received or a predefined waiting duration (DIFS 28) elapses, the synchronization adjustment process may only consists in going to step 605 in order to load the next smallest backoff value (at next occurrence of step 625). However, this may induce de-synchronization, in particular if no RTS message is ultimately received.

In this second situation where the hardware backoff counter reaches 0 while no other collaborative node accesses the communication medium or such a node accesses it with a significant delay, the local node also resynchronizes the current backoff value.
Advantageously, one can note the re-synchronization operation only involves modifying the current backoff counter value, and not to the whole list of backoff values in memory. The re-synchronization operation thus requires low processing resources.

Next to the re-synchronization operation, the process loops back to step 605 to generate a new backoff value to replace the one that has just elapsed at step 635. It means that the newly generated backoff value is associated with the same node as the backoff value that has just expired.

This loop back to step 605 makes it possible that upon detecting the expiry of one hardware backoff counter previously loaded with the backoff value associated with a given collaborative communication node, a new backoff value is generated for said given collaborative communication node.

It is worth noting that this embodiment allows the scheduling in advance of the next transmission period of the community. Thus, the backoff generation may be adapted according to the current values of the backoff list in order to achieve a kind of periodicity.

In the process shown in the Figure and described above, step 630 is performed before the backoff countdown procedure. In a variant, this step 630 may be performed after the backoff counter expires (i.e. after step 635). In that case, the backoff value to subtract has to be kept in the backoff list or in a dedicated place in memory.

Figure 8 illustrates the evolving of the backoff list as successive medium accesses are granted to the community of collaborative nodes.

The example of the Figure starts at the end of the initialization phase shown in Figure 7, when the minimum backoff value is set in the hardware backoff counter and the backoff list is updated (table 740).

When the hardware backoff counter expires, Node 1 which is the owner of the elapsed backoff is granted an access to the communication medium (TXOP can be shared with the whole community) and a new backoff value is generated for Node 1. This is a new occurrence of backoff generation step 605, shown in Figure 8 using arrow 810 between table 740 and table 820. It is recalled here that the random set obtained from the seed is the following: 15, 3, 8, 11, 1, 7 and 12, and that the values 15, 3, 8 and 11 have already been used during the initialization phase.

Thus, in this example, the new backoff value generated for Node 1 is '1'. It is inserted in the backoff list, and the ordering (620) and update (630) processes can
be performed. This situation is illustrated by table 820 showing that this new backoff value is the smallest one, and it is thus loaded (625) in the hardware backoff counter.

When the hardware backoff counter expires anew, Node 1 again accesses the communication medium (TXOP shared with the community) and a new backoff value is generated again for Node 1, having the value ‘7’ in the present example. The new occurrence of step 605 is shown through arrow 830 and results in obtaining table 840.

As the value ‘7’ is already present in the backoff list, a collision may occur between Nodes 1 and 3 if this value is kept. To avoid an actual collision, a new backoff is drawn for Node 1 instead of the previously generated value ‘7’ (arrow 850 represents the occurrence of step 615, resulting in obtaining a new backoff value ‘12’ for Node 1). As a result, in the backoff list 860, due to the ordering step, the smallest backoff value is ‘4’ associated with Node 2, which is thus loaded in the hardware backoff counter to obtain the next medium access for the community. The other backoff values in the list are decremented by the same amount (‘4’).

This first implementation (but also the other implementations as described below) of the invention shows that the latter is easy to implement, makes easier the management of the community to obtain medium accesses, and allows a dynamic node insertion/deletion (as number of nodes is independent to backoff resources).

For example, as a new node enters the community, all the collaborative nodes are warned of it at the next medium access by this new node. In response they all generate a backoff value for this new node (e.g. by taking the next value in the random set). This new value is added to the backoff list, and associated with the new node at the next backoff expiration of a community node. Upon the next medium access for the community, the current backoff list, the seed and the number of iteration may be sent to the new node for it to be fully synchronized with the community. A variant may consist to reset the random generator for all the nodes (i.e. to restart with the initial random sequence) when a new node is inserted in the community.

When a node leaves the community, its associated backoff value is deleted from the backoff list in each collaborative node, as soon as the latter is aware of such leaving.

Figure 9 illustrates a second implementation of the invention which is a variant of the first implementation (the local node still has only one hardware backoff counter). As for Figure 6, the process shown in this Figure is performed by each collaborative node.
While the first implementation is based on having in memory a backoff value for each collaborative node, this second exemplary implementation relies on having only one backoff value at the same time in the local node. To do so, the step of generating the backoff values comprises generating only one backoff value each time the hardware backoff counter expires. Then, the newly generated backoff value is associated with one of the collaborative communication nodes according to a predefined node order (e.g., Round-robin scheduling) of the collaborative communication nodes and is loaded in the hardware backoff counter to be decremented.

In other words, unlike the first implementation above in which one backoff value is always available (in the list) for each node of the community, the second implementation provides that only one backoff value is drawn at a given time and is assigned to a different node of the community (typically in a round robin way).

As for the first embodiment, the single hardware backoff counter is not used as a conventional local backoff counter (i.e., to trigger the sending of a RTS message or the like), but as a community backoff counter which loads and decrements backoff values that successively correspond to either the local node (thus to trigger such sending of a RTS message) or the other collaborative nodes (to monitor their respective backoff procedure countdown).

The process of this second implementation starts at step 900 similar to seed sharing step 600 described above.

Once the random generator of the local node is initialized with the common seed, the system resets a local indexing variable ‘count’ to 0 at step 905. This local variable ‘count’ enables the local node to know which collaborative node among the community should next access the medium.

Next, the process goes to step 910 where it generates a new backoff value (at this stage no other backoff value was already in memory of the local node). As the same seed is shared by all the nodes of the community, the backoff value generated by each collaborative node is the same. This backoff value is set in the hardware backoff counter at step 915.

Next step 920 of performing a countdown of the hardware backoff counter is performed as long as the communication medium is sensed idle and is suspended when a transmission is detected on the communication medium. This is similar to step 635 described above.
As long as the communication medium is sensed idle, the local node checks (step 920) whether or not the backoff time elapses, i.e. the hardware backoff counter expires by reaching 0.

When the communication medium becomes busy, the local node suspends the countdown and tests whether or not the data received over the communication medium come from one collaborative node of the community (test 925 similar to step 640). For example, the data received are generally a RTS message when the four-way handshaking mechanism is implemented.

If the local node receives a RTS message from a collaborative node while the backoff counter has not yet expired, a de-synchronization among the collaborative nodes is thus detected. The process thus goes to step 940 (similar to step 655) where a re-synchronization operation is performed. Next to the re-synchronization operation, the process goes to step 950 described below.

Otherwise (if the data received are from a legacy node – output “no” at step 925), the process loops back to step 920 to wait for the communication medium to be idle again in order to resume the backoff countdown procedure. Thus, the local node waits for backoff time expiration or for a new transmission from another node.

When the backoff counter expires, the process goes to step 930 to check whether or not the local node is the owner of the backoff value that has been decremented in the counter.

For instance, in Figure 9, the local node uses the local variable ‘count’ to specify with which node of the community the backoff value generated at step 910 and loading in the backoff counter at step 915 is associated (i.e. which node has to next attempt a medium access).

At step 930, the local node checks whether the variable ‘count’ is equal to its node identifier or to the identifier of another collaborative node (assuming that the node identifiers of the N collaborative nodes are from 0 to N-1).

If ‘count’ corresponds to the local node (test 930 is positive), it means that the local node is granted a medium access for a TXOP. As a consequence, the local node attempts to access the communication medium by sending a RTS message at step 945 (similar to step 660). A conventional communication using a TXOP is thus performed as described above with respect to Figure 3 before going to step 950.

Otherwise (‘count’ corresponds to another collaborative node – output “no” at step 930), the local node waits for a RTS message to be received from said other collaborative node. This is step 935 similar to step 650.
If the local node receives such a RTS message, the process then goes to step 950.

If the local node does not receive such a RTS message or receives it with a delay or time drift, it goes to step 940 to perform a re-synchronization operation (synchronization adjustment step).

The synchronization adjustment step is similar to step 655 described above. For example, the backoff counter is set to 0 upon receiving the message from the other collaborative node (lagged-behind situation); and any new backoff value that is loaded in the backoff counter after its expiry is loaded anew (with its original value) upon receiving the message from the other collaborative node (ahead situation).

At step 950, the process increments the variable 'count' to give the token to the next node of the community. This increase is performed with modulo the number of node in the community to ensure that each collaborative node is periodically considered. Next, the process loops back to step 910 to generate a new backoff value for the collaborative node having the token.

One may note that, in this second exemplary implementation, the successive backoff values are cumulated to form the waiting time each collaborative node has to wait before trying to access the communication medium. This is because, as the collaborative nodes are successively considered, they each waits its backoff time plus the backoff time assigned to each other node of the community previously considered.

For instance, if the first node of the community draws a backoff value equal to 2 and the second node a backoff value equal to 15, the second node of the community has to wait for 2+15 backoff slots before attempting to access the medium.

As a consequence, this second implementation may appear not to be fair for the community nodes in comparison to the legacy nodes following the 802.11 standard. It is recalled here that the backoff values (for both the collaborative nodes and the legacy nodes) are chosen randomly from a uniformly distributed random variable with an interval [0, CW], where CW (integer) is referred to as the contention window.

To improve the medium access for the collaborative nodes, the difference of fairness between the collaborative nodes and the legacy nodes should be reduced. This may be performed by decreasing the contention window size used for the collaborative nodes only, for example as a function of the number of collaborative nodes. Embodiments of the invention thus provide that the backoff values for the
collaborative nodes are generated using a uniformly distributed random variable with an interval [0, CCW], where CCW is the size of a collaborative contention window strictly less than the size of the contention window. In particular, CCW may be a function of the size of the contention window and of the number of collaborative communication nodes.

As an example, CCW may be equal to CW/N.

Additionally, the value CW for the community may be adapted dynamically, for example by correlating CW with the traffic activity of the community. For instance, when less traffic has to been conveyed for the community, CW is increased so that less medium accesses are performed by the community. As a result, CW and indirectly CCW may manage the community fairness with respect to legacy nodes.

Compared to the first implementation of the invention, the approach of Figure 9 provides additional advantages. In particular, as the process draws only one backoff value at a time, there is no risk of internal collision or backoff value duplication. Thus corresponding checks and correcting operations (e.g. step 615) are no longer needed.

Also, this approach prevents from any limitation on the number of collaborative nodes in the community. This is because the first implementation avoids duplication between the backoff values associated with the collaborative nodes, therefore limiting their number to CW+1. Here, only one backoff value is handled at a given time, avoiding the above limitation to apply.

Figure 10 illustrates a third exemplary implementation of the invention in which the local node has two or more hardware backoff counters to manage its own backoff value and the backoff values of the other collaborative nodes.

In this third exemplary implementation, the generated backoff value associated with the local node is loaded in a first hardware backoff counter of the local node to be decremented, and the other generated backoff values are successively loaded in at least one other hardware backoff counter of the local node.

The at least one other hardware backoff counter may be one, two or more, for example three because the nodes have usually at most four hardware backoff counters (one of which being dedicated to the local backoff value in this implementation of the invention).

The process described below with reference to Figure 10 provides advantages when the local node includes less backoff counters than the number of collaborative communication nodes in the community.
In the implementation of the Figure, the other hardware backoff counter or counters are managed more or less as described above with reference to Figures 6 to 8, i.e. the counters, upon expiry, are successively assigned to the ascending backoff values of the other collaborative nodes.

In a variant, the approach of Figure 9 could be used to handle the other hardware backoff counter or counters.

The process starts at step 1000 by the generation of the backoff list (for instance for a community of N nodes). This step 1000 groups steps 605, 610 and 615 of Figure 6.

At the end of step 1000, a backoff list of N backoff values arranged in ascending order has been produced without duplicate values. Each backoff value is associated with a distinct collaborative node of the community.

Next, the process goes to step 1005 in which one hardware backoff counter called local backoff counter is set with the own backoff value associated with the local node, i.e. it is set with the backoff value the expiration of which triggers the medium access by the local node (in the contrary, the local node has to cancel the automatic medium access request for the other backoff counters used to monitor the backoff procedures of the other collaborative nodes).

Next to step 1005, the process goes to step 1010 to check whether or not one other hardware backoff counter is available, i.e. not already loaded with a non-zero backoff value. If one hardware backoff counter is available, the local node sets this hardware backoff counter with the next smallest backoff value of the backoff list. This is step 1015 that loops back to step 1010.

Steps 1010 and 1015 in association allow an assignment of each available hardware backoff counter to be performed with a distinct backoff value of the backoff list.

When all the hardware backoff counters available on the local node are used (output "no" at step 1010), the process goes to step 1020 to update the remaining backoff values of the backoff list in the same way as step 630 above.

For instance, this update consists to subtract a determined value from all the backoff values of the list, the determined value being equal to the smallest backoff non-zero value between the backoff value or values loaded in the other hardware backoff counters during the last series of loops 1010-1015 and the backoff value already present in the other hardware backoff counters at the beginning of this last series.
During an initialization phase (e.g. when starting the process of Figure 10), it is the smallest backoff value that is loaded in the other hardware backoff counters during the last series of loops 1010-1015. This is because all the other hardware backoff counters were available (i.e. without value loaded therein) at the beginning of this last series.

When there is only one "other" hardware backoff counter, the determined value is the smallest backoff value that is loaded in the single other hardware backoff counter.

In stable state (i.e. not during an initialization phase) where only one other backoff counter is loaded with a new backoff value during the last series of loops 1010-1015, the determined value should be either the new backoff value that is loaded (in case it is lower than all the values currently in the other hardware backoff counters), or the smallest non-zero backoff value already present in the other hardware backoff counters at the beginning of this last series (in case the backoff value newly loaded is higher than this smallest non-zero backoff value).

Note that no update of the backoff list is performed when a backoff value is only loaded in the local hardware backoff counter (this occurs on a new occurrence of steps 1000-1010 after the local counter has expired).

The new backoff values that result correspond to their current values when the next other hardware backoff counter will expire.

Next to the update 1020, the backoff countdown procedure of all the hardware backoff counters is performed as long as the communication medium is sensed idle and is suspended when a transmission is detected on the communication medium. This is step 1025 similar to steps 635 and 920 described above.

As long as the communication medium is sensed idle, the local node checks (step 1025) whether or not one of the loaded backoff times elapses, i.e. one of the hardware backoff counters expires by reaching 0.

When the communication medium becomes busy, the local node suspends the countdown and tests whether or not the data received over the communication medium come from one collaborative node of the community (test 1030 similar to steps 640 and 925). For example, the data received are generally a RTS message when the four-way handshaking mechanism is implemented.

If the local node receives a RTS message from a collaborative node while the associated backoff counter has not yet expired, a de-synchronization among the collaborative nodes is thus detected. The process thus goes to step 1045 (similar to
steps 655 and 940) where a re-synchronization operation is performed. Next to the re-
synchronization operation, the process loops back to step 1000.

Otherwise (if the data received are from a legacy node – output “no” at step 1030), the process loops back to step 1025 to wait for the communication medium to
be idle again in order to resume the backoff countdown procedure. Thus, the local
node waits for backoff time expiration or for a new transmission from another node.

When one of the hardware backoff counters expires, the process goes to step 1035 to check whether or not it is the local backoff counter or one of the “other”
backoff counters

If it is the local backoff counter, it means that the local node is granted a
medium access for a TXOP. As a consequence, the local node attempts to access the
communication medium by sending a RTS message at step 1050 (similar to steps 660
and 945). A conventional communication using a TXOP is thus performed as described
above with respect to Figure 3 before looping back to step 1000.

If it is one of the other hardware backoff counters, the local node waits for a
RTS message to be received from the associated collaborative node. This is step 1040
similar to steps 650 and 935.

If the local node receives such a RTS message, the process then loops
back to step 1000.

If the local node does not receive such a RTS message or receives it with a
delay or time drift, it goes to step 1045 to perform a re-synchronization operation
(synchronization adjustment step).

The synchronization adjustment step is similar to steps 655 and 940
described above. For example, the “other” hardware backoff counter decrementing the
backoff value associated with a collaborative node is set to 0 upon receiving the
message from this collaborative node (lagged-behind situation), and the other backoff
counters (including the local one) are reduced by the same amount, called time drift
(equal to the value of the “other” hardware backoff counter upon receiving the
message).

Also, in case a new backoff value is loaded in an “other” hardware backoff
counter that has expired, this new backoff value is loaded anew (with its original value)
upon receiving a message from the other collaborative node the associated backoff
value of which has elapsed in the “other” hardware backoff counter that has expired
(ahead situation). Again, all the other backoff counters (including the local one) are
modified by the same amount, call time drift (equal to difference between the original value of the new backoff value and its current value upon receiving the message).

After the synchronization adjustment step, the process loops back to step 1000 to draw a new backoff value for the collaborative node (including the local node), the associated backoff value has just elapsed.

As one may easily understand from the above, if this new backoff value is for the local node, it is directly loaded in the local backoff counter. And if it is for another collaborative node, it is added to the backoff list, the smallest value of which is thus loaded in the available backoff counter dedicated to monitor the other collaborative nodes.

Although the present invention has been described hereinabove with reference to specific embodiments, the present invention is not limited to the specific embodiments, and modifications which lie within the scope of the present invention will be apparent to a person skilled in the art. Many further modifications and variations will suggest themselves to those versed in the art upon making reference to the foregoing illustrative embodiments, which are given by way of example only and which are not intended to limit the scope of the invention as determined by the appended claims. In particular different features from different embodiments may be interchanged, where appropriate.
CLAIMS

1. A method of contending for access to a communication medium shared between communication nodes, comprising at a first communication node:
   generating backoff values driving backoff procedures of a Collision Avoidance mechanism for a plurality of collaborative communication nodes including the first communication node, each backoff value being associated with a respective collaborative communication node;
   loading at least one hardware backoff counter with the backoff values, each hardware backoff counter being decremented over time until expiry before loading a new backoff value therein; and
   upon detecting the expiry of one of the hardware backoff counters,
   accessing the communication medium if the backoff value loaded in the hardware backoff counter that has just expired is associated with the first communication node, and loading a new backoff value in the hardware backoff counter that has just expired,
   otherwise, directly loading the hardware backoff counter that has just expired with the backoff value associated with a collaborative communication node different from the one with which the backoff value last loaded in that hardware backoff counter is associated.

2. The method of Claim 1, wherein the generated backoff values are successively loaded in one and the same hardware backoff counter of the first communication node, and the method further comprising, upon detecting the expiry of the hardware backoff counter, determining whether or not the backoff value last loaded in the hardware backoff counter that has just expired is associated with the first communication node.

3. The method of Claim 2, wherein the step of generating the backoff values comprises generating only one backoff value each time the hardware backoff counter expires, and
   wherein the newly generated backoff value is associated with one of the collaborative communication nodes according to a predefined node order of the collaborative communication nodes and is loaded in the hardware backoff counter to be decremented.

4. The method of Claim 3, wherein the first communication node comprises only one backoff value at the same time.
5. The method of Claim 1, wherein the generated backoff value associated with the first communication node is loaded in a first hardware backoff counter of the first communication node to be decremented, and the other generated backoff values are successively loaded in at least one other hardware backoff counter of the first communication node.

6. The method of Claim 5, wherein the first communication node includes less backoff counters than the number of collaborative communication nodes.

7. The method of Claim 5 or 6, wherein the step of generating the backoff values comprises generating only one backoff value each time a hardware backoff counter expires, and

wherein if the hardware backoff counter that has just expired is the first hardware backoff counter, the newly generated backoff value is associated with the first communication node and is loaded in the first hardware backoff counter,

otherwise, the newly generated backoff value is associated with one of the other collaborative communication nodes according to a predefined node order and is loaded in the hardware backoff counter that has just expired.

8. The method of Claim 7, wherein the first communication node comprises, at any time, as many backoff values as the number of hardware backoff counters, but not more.

9. The method of Claim 1, wherein, upon detecting the expiry of one hardware backoff counter previously loaded with the backoff value associated with a given collaborative communication node, a new backoff value is generated for said given collaborative communication node.

10. The method of Claim 9, wherein the hardware backoff counter that has just expired is loaded with the smallest backoff value from amongst the newly generated backoff value and the backoff values associated with the other collaborative communication nodes.

11. The method of Claim 10, wherein the smallest generated backoff value so loaded is subtracted from each remaining backoff value not yet loaded, and the resulting backoff values are stored in a memory of the first communication node.

12. The method of Claim 1, wherein non-collaborative communication nodes access the communication medium based on a contention window, and the backoff values for the collaborative nodes are generated using a uniformly distributed random variable with an interval [0, CCW], where CCW is the size of a collaborative contention window strictly less than the size of the contention window.
13. The method of Claim 12, wherein CCW is a function of the size of the contention window and of the number of collaborative communication nodes.

14. The method of Claim 1, further comprising, at the first communication node, performing a synchronization adjustment process of the one or more hardware backoff counters upon receiving a message from another collaborative communication node at a time drifted from the expiry of the backoff value associated with said other collaborative communication node.

15. The method of Claim 14, wherein the synchronization adjustment process comprises subtracting a time drift from the one or more hardware backoff counters when the message is received before the expiry of the hardware backoff counter decrementing the backoff value associated with said other collaborative communication node, the time drift being equal to the backoff value associated with said other collaborative communication node that is being decremented when receiving the message.

16. The method of Claim 14, wherein if the message is received with a time drift after the expiry of the hardware backoff counter decrementing the backoff value associated with the other collaborative communication node having sent the message and a new backoff value has been loaded in the same hardware backoff counter after the expiry of the hardware backoff counter, the synchronization adjustment process comprises, upon receiving the message, resetting the hardware backoff counter with said new backoff value as previously loaded.

17. The method of Claim 15 or 16, further comprising modifying any other hardware backoff counter by the same amount of time.

18. A non-transitory computer-readable medium storing a program which, when executed by a microprocessor or computer system in a communication device sharing a communication medium with other communication nodes, causes the communication device to perform contention for access to the communication medium, including the steps of:

   generating backoff values driving backoff procedures of a Collision Avoidance mechanism for a plurality of collaborative communication nodes including the first communication node, each backoff value being associated with a respective collaborative communication node;

   loading at least one hardware backoff counter with the backoff values, each hardware backoff counter being decremented over time until expiry before loading a new backoff value therein; and
upon detecting the expiry of one of the hardware backoff counters,
accessing the communication medium if the backoff value loaded in the
hardware backoff counter that has just expired is associated with the first
communication node, and loading a new backoff value in that hardware backoff counter
that has just expired,
otherwise, directly loading the hardware backoff counter that has just expired with the backoff value associated with a collaborative communication node
different from the one with which the backoff value last loaded in that hardware backoff
counter is associated.

19. A device comprising means adapted for carrying out each step of the
method according to Claim 1.

20. A communication device contending for access to a communication
medium shared between communication nodes, the communication device comprising:
a backoff generator configured for generating backoff values driving backoff
procedures of a Collision Avoidance mechanism for a plurality of collaborative
communication nodes including the communication device, each backoff value being
associated with a respective collaborative communication node;
a backoff loader configure for loading at least one hardware backoff counter
with the backoff values, each hardware backoff counter being decremented over time
until expiry before loading a new backoff value therein; and
the communication device being configured, upon detecting the expiry of
one of the hardware backoff counters, for:
accessing the communication medium if the backoff value loaded in the
hardware backoff counter that has just expired is associated with the communication
device, and loading a new backoff value in the hardware backoff counter that has just
expired,
otherwise, directly loading the hardware backoff counter that has just expired with the backoff value associated with a collaborative communication node
different from the one with which the backoff value last loaded in that hardware backoff
counter is associated.

21. The communication device of Claim 20, comprising a single hardware
backoff counter to successively load the generated backoff values.

22. The communication device of Claim 21, wherein the backoff loader is
configured for associating a backoff value newly generated by the backoff generator
upon expiry of the hardware backoff counter with one of the collaborative
communication nodes according to a predefined node order of the collaborative communication nodes and for loaded the newly generated backoff value in the hardware backoff counter to be decremented.

23. The communication device of Claim 20, comprising a first hardware backoff counter to store and decrement the generated backoff value associated with the communication device, and at least one other hardware backoff counter to successively store and decrement the other generated backoff values.

24. The communication device of Claim 20, wherein the backoff generator is configured for, upon detecting the expiry of one hardware backoff counter previously loaded with the backoff value associated with a given collaborative communication node, generating a new backoff value for said given collaborative communication node.

25. The communication device of Claim 24, wherein the backoff loader is configured for loading, in the hardware backoff counter that has just expired, the smallest backoff value from amongst the newly generated backoff value and the backoff values associated with the other collaborative communication nodes.

26. The communication device of Claim 25, configured for subtracting the smallest generated backoff value so loaded from each remaining backoff value not yet loaded, and for storing the resulting backoff values in a local memory.

27. The communication device of Claim 20, further comprising a synchronization adjustment module configured for performing a synchronization adjustment process of the one or more hardware backoff counters upon receiving a message from another collaborative communication node at a time drifted from the expiry of the backoff value associated with said other collaborative communication node.

28. The communication device of Claim 27, wherein the synchronization adjustment module is configured for subtracting a time drift from the one or more hardware backoff counters when the message is received before the expiry of the hardware backoff counter decrementing the backoff value associated with said other collaborative communication node, the time drift being equal to the backoff value associated with said other collaborative communication node that is being decremented when receiving the message.

29. The communication device of Claim 27, wherein if the message is received with a time drift after the expiry of the hardware backoff counter decrementing the backoff value associated with the other collaborative communication node having sent the message and a new backoff value has been loaded in the same hardware
backoff counter after the expiry of the hardware backoff counter, the synchronization adjustment module is configured for, upon receiving the message, resetting the hardware backoff counter with said new backoff value as previously loaded.

30. The communication device of Claim 28 or 29, wherein the synchronization adjustment module is configured for modifying any other hardware backoff counter by the same amount of time.
Fig. 2

TXOP

Timer resumed and then re-init

Timer suspended

Backoff countdown (access contention)

Backoff countdown (access contention)

Defer access

Source

Destination

Others

NAV (RTS)

NAV (CTS)

SIFS

DIFS

ACK

DATA

RTS

CTS

T0

T1

T2

T3

20

21

22

23

24

25

26

27

28

29

30

31

240

220

230

27

28

260

210

27

255

250

32

270

270
Fig. 3

Backoff slots (260): DIFS

Transmission

ACK

Busy NAV

Collaborative group backoff positioning

Nth backoff slots 260:

n+1, n+2, n+3

n, n+4, n+5, n+6, n+7, n+8

n+9, n+10, n+11, n+12, n+13, n+14

DIFS

SIFS

28

250

28

230

27

240

330

300

320

310
Fig. 6

1. Send seed to all group nodes (600)
2. Generate backoff values per node (610)
3. Check duplicate value (615)
4. Backoff list ordering (620)
5. Check if backoff time elapsed (630)
6. Set backoff counter with min backoff value (625)
7. Update backoff list (630)
8. Send RTS (660)
9. Syncro adjustment (655)
10. RTS from a collab. node? (640)
   - Yes, is it my turn? (645)
   - Yes, send RTS (660)
   - No, wait for RTS from a collab. node (640)
   - No, wait for RTS from a collab. node (640)
   - No, wait for RTS from a collab. node (640)
   - No, wait for RTS from a collab. node (640)
### Group Backoff Value

<table>
<thead>
<tr>
<th>Transmission Node</th>
<th>Group Backoff Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node 1</td>
<td>1 -&gt; 0</td>
</tr>
<tr>
<td>Node 2</td>
<td>5 -&gt; 4</td>
</tr>
<tr>
<td>Node 3</td>
<td>8 -&gt; 7</td>
</tr>
<tr>
<td>Node 0</td>
<td>12 -&gt; 11</td>
</tr>
</tbody>
</table>

### Group Backoff Value

<table>
<thead>
<tr>
<th>Transmission Node</th>
<th>Group Backoff Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node 2</td>
<td>4 -&gt; 0</td>
</tr>
<tr>
<td>Node 3</td>
<td>7 -&gt; 3</td>
</tr>
<tr>
<td>Node 0</td>
<td>11 -&gt; 7</td>
</tr>
<tr>
<td>Node 1</td>
<td>12 -&gt; 8</td>
</tr>
</tbody>
</table>

**New Bch Generated for Node 1 (1)**

**New Bch Generated for Node 1 (7)**

**New Bch Generated for Node 1 (12)**

---

### Group Backoff Value

<table>
<thead>
<tr>
<th>Transmission Node</th>
<th>Group Backoff Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node 1</td>
<td>3 -&gt; 0</td>
</tr>
<tr>
<td>Node 2</td>
<td>8 -&gt; 5</td>
</tr>
<tr>
<td>Node 3</td>
<td>11 -&gt; 8</td>
</tr>
<tr>
<td>Node 0</td>
<td>15 -&gt; 12</td>
</tr>
</tbody>
</table>

### Group Backoff Value

<table>
<thead>
<tr>
<th>Transmission Node</th>
<th>Group Backoff Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node 2</td>
<td>4</td>
</tr>
<tr>
<td>Node 3</td>
<td>7</td>
</tr>
<tr>
<td>Node 1</td>
<td>7</td>
</tr>
<tr>
<td>Node 0</td>
<td>11</td>
</tr>
</tbody>
</table>
Fig. 9

900
Send seed to all group nodes

905
Count = 0

910
Generate new backoff value

915
Set backoff counter with backoff value

920
Backoff time elapsed?

925
RTS from a collab node?

930
Count = my node id?

935
no

940
Syncro adjustment

945
Send RTS

950
Count++ (mod number of node in the group)
**INTERNATIONAL SEARCH REPORT**

**A. CLASSIFICATION OF SUBJECT MATTER**

**INV. H04W74/08**

**ADD.**

According to International Patent Classification (IPC) or to both national classification and IPC

**B. FIELDS SEARCHED**

Minimum documentation searched (classification system followed by classification symbols)

H04W

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal, WPI Data

**C. DOCUMENTS CONSIDERED TO BE RELEVANT**

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>GB 2 490 963 A (CANON KK [JP]) 21 November 2012 (2012-11-21) cited in the application page 5, line 20 - page 6, line 3 page 7, line 31 - page 8, line 3 page 8, line 30 - page 9, line 22 page 13, line 29 - page 17, line 5 figure 5</td>
<td>1-5,7-30</td>
</tr>
<tr>
<td>Y</td>
<td>GB 2 502 576 A (CANON KK [JP]) 4 December 2013 (2013-12-04) page 5, line 20 - page 9, line 6</td>
<td>14-17, 27-30</td>
</tr>
</tbody>
</table>

☐ Further documents are listed in the continuation of Box C. ☑ See patent family annex.

* Special categories of cited documents:

- **A** document defining the general state of the art which is not considered to be of particular relevance
- **E** earlier application or patent but published on or after the international filing date
- **L** document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
- **O** document referring to an oral disclosure, use, exhibition or other means
- **P** document published prior to the international filing date but later than the priority date claimed

- **T** later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

- **X** document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

- **Y** document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

- **B** document member of the same patent family

**Date of the actual completion of the international search** 11 August 2015

**Date of mailing of the international search report** 17/08/2015

**Name and mailing address of the ISA/European Patent Office, P.B. 5018 Patentlaan 2 NL-2280 HV Rijswijk Tel. (+31-70) 340-3040, Fax: (+31-70) 340-3016**

Hegeman, Hans
<table>
<thead>
<tr>
<th>Patent document cited in search report</th>
<th>Publication date</th>
<th>Patent family member(s)</th>
<th>Publication date</th>
</tr>
</thead>
<tbody>
<tr>
<td>GB 2490963 A</td>
<td>21-11-2012</td>
<td>GB 2490963 A</td>
<td>21-11-2012</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2014153514 A1</td>
<td>05-06-2014</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WO 2012160474 A1</td>
<td>29-11-2012</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2003145095 A1</td>
<td>31-07-2003</td>
</tr>
<tr>
<td>GB 2502576 A</td>
<td>04-12-2013</td>
<td>GB 2502576 A</td>
<td>04-12-2013</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2014003367 A1</td>
<td>02-01-2014</td>
</tr>
</tbody>
</table>