Abstract—In a Beaconing Scheme (BS) each network node periodically broadcasts messages (beacons) to obtain and maintain an image of the network physical topology. Node’s mobility and issues related with the communication process could degrade the BS view accuracy. This work proposes a novel method to improve the BS robustness face to the communication process drawbacks. Assuming a homogeneous traffic load, each node uses the information given by the medium access control (MAC) layer to dynamically adapt the Beaconing Timeout value. Several simulation results evaluate the adaptive method performance using the IEEE 802.11b DCF protocol. We conclude that the BS accuracy can be improved even when node’s mobility is assumed.  

I. INTRODUCTION

In the recent years a strong research effort has been made to propose and deploy routing schemes for Mobile Ad Hoc Networks (MANETs). MANETs do not have a fixed infrastructure. They often present a high degree of node’s mobility causing rapid changes in the topology. Routing Protocols (RPs) like ABR [1], OLSR [2] or VRR [3], periodically broadcast messages (beacons) to obtain and maintain the network topology. The received beacons are used to learn about each node’s neighborhood. If two nodes are within range, a physical link exists and the beaconing scheme (BS) helps to detect it. But the BS is based on a node’s neighbor polling mechanism, which induces some mismatches. The instant when a physical link is established does not exactly match with the instant when it is firstly detected by the polling mechanism. When a physical link breaks, it follows the same time mismatch. The inaccurate information about the physical links justifies the notion of virtual link - the link information that is possible to obtain with a BS.

The absence of beacons received by a node from one of its neighbors during a pre-determined time (Beaconing Timeout) signals that the link with that neighbor is broken. A node does not receive a beacon from one of its neighbor nodes if:

1) one of the two nodes moves away, being out-of-range;
2) the neighbor node is turned off;
3) the beacon sent is lost in the communication process (due to a collision);

While the first two reasons are related with a network topology change, the third one is related with the medium access control (MAC) protocol used. For the first two reasons, when the beaconing timeout expires the BS assumes a broken virtual link. In this case, a lower beaconing timeout minimizes the BS time needed to perceive the physical link lost. For the third reason a virtual link can be considered broken if a beacon is transmitted but not received. But now the physical link still exists. In this case, the probability of erroneously considering the link broken increases if a smaller timeout value is adopted. Thus, the beacon timeout value \(T_0\) is a critical BS parameter. It trades off between the node’s mobility and the MAC reliability assumed.

Although many works use BSs, to the best of our knowledge only [5] and [6] addresses the BS parametrization problem. [5] presents a routing protocol for wireless networks where a BS is used. This work defines an inter-beacon transmission interval with average \(B\), presenting the routing protocol packet delivery ratio of a mobile network for distinct beacon transmission rates. But the beacon transmission rate (BTR) and the beaconing timeout value \((4.5B)\) adopted were not justified. The work presents the routing behavior as function of BTR rates, concluding that lower rates decrease the protocol's packet delivery as more inaccurate virtual links exists due to the node’s mobility. [6] proposes and analyzes mechanisms to improve the accuracy of virtual links in BSs. The authors assume that the position information is available at each node. They determine the probability of a node B being out of transmission range of a given node A, after a given time interval, given the probability density function (PDF) of a node’s speed. The work addresses the first reason of beacon loss (due to mobility) justifying the beaconing timeout interval when distinct average node’s speeds are used. The expected percentage of outdated virtual links as function of distinct beaconing timeouts is plotted, concluding that larger beaconing timeout values decrease the virtual links accuracy as the average node’s speed increases.

The beacons lost due to the third beacon loss factor have also a great impact in the virtual links accuracy. Our work assumes the IEEE 802.11b protocol. The 802.11 broadcast transmission scheme that supports the beacons transmission is unreliable [9]: several collisions can occur and the unsuccessful broadcast transmissions are not retransmitted. Moreover, it does not provide the necessary quality of service to guarantee a periodical message delivery, as the channel access delay depends on its occupancy. The combinations of these two effects can origin erroneous behaviors in the routing algorithm, because it may consider broken virtual links that are physically established.

This work proposes a variable beaconing timeout scheme to increase the BS accuracy. The beaconing timeout is changed according to the information collected at the MAC layer. As far as we know, this is the first study that proposes an adaptive runtime beaconing timeout, and an analysis of the MAC’s
The BS is explained in detail in the following section. The IEEE 802.11 MAC features are introduced in the section III, where the adaptive beaconing timeout method is explained. Some examples are given from simulation results obtained with a motionless fully connected network. Section IV analyzes several simulation results considering node’s mobility. It is shown that the adaptive beaconing timeout could improve the BS accuracy. Finally, a few concluding remarks are given in Section V.

II. PROBLEM OVERVIEW

In a Beaconing Scheme a node periodically broadcasts a Beacon. Fig. 1 depicts the time diagram of a BS. The inter-beacon transmission period is denoted by $T_B$. The beacon sent by the node A reach the node B after a

$$T_B = T_{MAC} + T_{PROP}$$  \hspace{1cm} (1)$$

time period, where $T_{PROP}$ models the transmission channel propagation delay and $T_{MAC}$ is the sum of the MAC transmission queue waiting time, the MAC contention to access the channel and the effective time to copy the entire beacon to the transmission channel. When the first beacon is received at node B, it creates a new entry on its neighborhood table with the virtual link signaled by the beacon reception. At the same instant, the node B sets up a timer with the beaconing timeout value

$$T_O = \beta T_B, \beta > 1.$$ \hspace{1cm} (2)$$

The virtual links exist as long as the timer does not expire between consecutive beacon arrivals. If the timer expires, the virtual link is considered broken and it is removed from the table. If the node A moves out of range of node B, the node B will not receive the beacon and the virtual link is considered broken (this situation was addressed in [6]). Fig. 2 depicts the virtual links duration time’s probability density function for a full connected motionless network with ten nodes. The results, obtained with the ns-2 simulator [8], were collected during a simulation time of 500 seconds. The 802.11b protocol was simulated assuming the lowest data bit rate of 1Mbps. The beacon transmission period ($T_B$) was set to 1 second, and the timeout was set to $T_O = 2.5T_B$. The parameter $\beta = 2.5$ was kept constant. Fig. 2 a) shows the simulation results when the network only has the BS traffic (beacons). The results show that the BS correctly maps all physical links with the virtual links.

Fig. 2 b) shows results when other traffic exists on the network. Now each node generates traffic according to a exponential distribution to achieve 40% of the network’s capacity usage. The traffic is composed by 10% of broadcast traffic and the remaining unicast. The results depicted in Fig. 2 b) show that the virtual links obtained through the BS do not match well with the physical links. While the physical links last 500 seconds, the virtual links last less. With $T_O = 2.5T_B$, if at least two consecutive beacons are lost due to collisions the timeout of a beacon’s receiver node expires without receiving a beacon. The same happens when multiple bursts of frames are queued to be transmitted on a near saturation network, which delays the beacon effective transmission. Fig. 2 c) show that

[Figures and tables are not transcribed here due to the nature of the task.]
the BS inaccuracy increases for saturated traffic condition.

The previous examples show that a constant \( \beta = 2.5 \) parameter could lead to highly inaccurate virtual links. Without node’s mobility this is due to the 802.11 MAC properties. Thus, a virtual link could undesirably be considered broken if the network suddenly works near the saturation \( T_{\text{MAC}} \) increases. Formally, the link is considered broken if

\[
\beta < \frac{T_b}{T_B}.
\]

This work focuses the abnormal situation when the MAC delays the beacon transmission and, although the physical link still exists, the virtual link is considered broken because of the \( \beta \) chosen value. In the following section, a novel method is described to obtain an adaptive value for \( \beta \).

III. ADAPTIVE BEACON TIMEOUT

A runtime adaptive value for \( \beta \) must be provided to avoid falling into the condition (3). When the node A sends a frame (it sends at least a beacon frame every \( T_B \) seconds) it can measure the channel conditions. The node A can thus estimate the \( T_{\text{MAC}} \) value for each future frame transmissions. As bidirectional physical links are considered between two nodes A and B, and because node’s homogeneous traffic is assumed, the \( \beta \) parameter of node A could be dynamically adapted using its own \( T_{\text{MAC}} \) estimate. In the following subsection an 802.11 MAC delay model is briefly introduced in order to obtain the \( T_{\text{MAC}} \) estimate.

A. MAC delay model

The model proposed in [4] is used to estimate \( T_{\text{MAC}} \). [4] presents a IEEE 802.11 delay model assuming both unicast and broadcast traffic in saturated and non-saturated conditions. It assumes a single-hop 802.11 MAC network, and derives the MAC frame’s service time for non-saturated traffic. The MAC transmission buffers are modeled using an M/M/1/K queueing model, which allows the characterization of the time \( T_{\text{MAC}} \) needed to transmit each frame, from the instant it is included in the MAC buffer until the end of its transmission.

When one of the \( n \) network nodes has a frame to transmit it waits during a random amount of time after which the frame transmission occurs. The random waiting time \( (T_C) \) improves the medium access fairness and is obtained through a slotted Binary Exponential Backoff (BEB) technique. The time is slotted and a backoff counter is initialized with a random number of states (backoff states) that are uniformly chosen from the interval \([1; W_i]\) (\( i \) models the distinct backoff stages considered). The channel is sensed in the beginning of every backoff state. The backoff counter decrements its state and waits a slot time when the channel is sensed idle. When the channel is initially found busy, the backoff state is frozen and the backoff node periodically senses the channel for the subsequent time slots. When the channel goes idle again for at least a constant amount of time [7], the backoff counter decrements again its backoff state. The transmission occurs when the backoff counter reaches the state zero.

From [4], the MAC service time is

\[
T_S = p_{\text{QE}}(\text{DIFS} + T_{C_1}) + E[l] + T_{C_1} + T_{C_r} - p_B T_{C_r}, \tag{4}
\]

where DIFS is the constant duration defined in the IEEE 802.11 standard [7], \( p_B \) is the probability of a node generating a broadcast frame, \( p_{\text{QE}} \) is the probability of finding the MAC queue empty and \( E[l] \) is the expected time needed to send the frame to the channel (for a fixed data transmission rate it depends on the frame size). Let \( T_S \) be the average time waiting on each backoff state. Because unicast transmissions could be retransmitted \( m - 1 \) times, while broadcast transmissions do not, \( T_{C_1} \) and \( T_{C_r} \), denote the average contention time spent in the first transmission or in unicast retransmissions, respectively [4]:

\[
T_{C_1} = \frac{W_1 - 1}{2} T_{\chi} \tag{5}
\]

\[
T_{C_r} = \sum_{i=2}^{m} \left[ 1 - (1 - \chi)^{i-1} \right] \left( \frac{W_i - 1}{2} T_{\chi} + E[l] \right). \tag{6}
\]

In (6) \( \chi \) is the probability of a node transmitting at a randomly chosen time slot, given by [4]:

\[
\chi = \frac{p_B + (1 - p_B) \frac{1 - a^m}{a}}{W_1 + 1 + (1 - p_B) \sum_{i=2}^{m} \frac{W_i + 1}{2} (1 - a^{i-1}) + b}. \tag{7}
\]

Assuming that \( n \) and \( p_B \) are known, \( p_{\text{QE}} \) can be obtained from [4]:

\[
p_{\text{QE}} = \frac{1 - \lambda T_S}{1 - (\lambda T_S)^{K + 1}}. \tag{8}
\]

\( K \) denotes the MAC queue size and \( \lambda \) the frame generation rate per node. Replacing (7) in (6), (5), (6) and (8) in (4), \( T_S \) can be numerically computed knowing \( n, \lambda, p_B, \) and \( T_{\chi} \). The average queueing MAC time \( T_q \) can be computed having \( T_S \) value and using [4]:

\[
T_q = \frac{\rho \left( 1 - (K + 1) \rho^K + K \rho^{K+1} \right)}{(1 - \rho) (1 - \rho^{K+1})} + p_{\text{QE}} - 1. \tag{9}
\]

Generalizing, the MAC overhead time \( T_{\text{MAC}} = T_s + T_q \) can be computed given the parameters \( n, \lambda, p_B, \) and \( T_{\chi} \):

\[
T_{\text{MAC}} = f(n, \lambda, p_B, T_{\chi}). \tag{10}
\]

The parameters \( a \) and \( b \) are respectively defined as \( a = (1 - \chi)^{n-1} \) and \( b = \frac{1}{1 - p_{\text{QE}}} \).

\( \rho \) expresses the traffic intensity, which is given by \( \rho = \lambda T_S \).
B. Estimation of MAC’s overhead time - $T_{MAC}$

The parameters $n$, $\lambda$, $p_b$, and $T_\chi$ must be estimated to use the equation (10) in runtime. Since [4] is valid for a single-hop network, a node must estimate the $n-1$ number of neighbors found 1-hop way from it. The $n$, $\lambda$ and $p_b$ estimates can be directly measured by each node by simply monitoring the channel activity. The channel is monitored during the last $T_{est}$ time period. During the $T_{est}$ time, if $k_b$ broadcast frames are observed in $k_T$ total frames, the estimate $\hat{p}_b$ is obtained through a simple average:

$$\hat{p}_b(t+1) = \frac{k_b}{k_T}, \quad (11)$$

The frame’s generation rate $\lambda$ per node in $T_{est}$ time interval is estimated using

$$\hat{\lambda}(t+1) = \frac{k_T}{(\hat{n} - 1)T_{est}}, \quad (12)$$

because homogeneous load is considered. The $\hat{n} - 1$ estimate is obtained counting the number of distinct source nodes that transmit a frame during the last $T_{est}$ time.

To estimate $T_\chi$, each node considers the last $q$ backoff stages applied. Underline that the $q$ backoff stages can be applied to a single or multiple frame transmissions, depending on the $q$ value chosen and on the number of retransmissions applied on each effective frame’s transmission. For the backoff stage $d$, $J_d$ backoff states are randomly chosen from the $[1, W]_{T_d}$ backoff window interval. Let’s assume $T_{\chi,d}$, the time spent in the backoff state $j$ occurring in the $d$ backoff stage. The average time spent in the backoff stage $d$ is given by:

$$\overline{T_d^\chi} = \frac{1}{J_d} \sum_{h=1}^{J_d} T_{\chi,d}^h. \quad (13)$$

The $T_\chi$ estimate is obtained applying a Weighted Moving Average filter:

$$\hat{T}_\chi(t+1) = \alpha\overline{T_d^\chi} + \frac{1 - \alpha}{q-1} \sum_{d=2}^{q} \overline{T_d^\chi}, \quad (14)$$

$\alpha$ is the filter weight. Note that $\overline{T_d^\chi}$ is the average time spent in the last backoff stage occurred.

Fig. 3 b) shows the $T_\chi$ estimate for a motionless network using 802.11b imposing distinct frame generation rates. The results were obtained using the ns-2 simulator [8]. From 0 to 250 seconds (period A) each node generates 30 frames per node per second, from 250 to 500 (period B) $\lambda$ equals to 45, from 500 to 750 (period C) $\lambda$ equals to 20 and finally, in the remaining time (period D) every node generate 50 frames per second. A fully connected network with 10 nodes is assumed. All nodes generate frames with variable size sampled from a uniform distribution in the interval [1, 2300], with an average of 1150 bytes. Some other simulation parameters are summarized in table 1. The parameters $q = 20$ and $\alpha = 0.1$ were assumed for (14). $T_{est} = T_B$ was considered to determine $\hat{n}$, $\hat{\lambda}$ and $\hat{p}_b$. From Fig. 3 it can be observed that using this parametrization $T_\chi$ behaves like a low pass filter of $T_X$. The four distinct frame’s generation rates are observed by the envelop offset. Fig. 3 also depicts (represented in white) the $T_\chi$ average in steady state, obtained with [4].

Having $\hat{n}(t+1)$, $\hat{\lambda}(t+1)$, $\hat{p}_b(t+1)$ and $\hat{T}_\chi(t+1)$, from (10) the estimate of $T_{MAC}$ is obtained applying:

$$\hat{T}_{MAC}(t+1) = f(\hat{n}(t+1), \hat{\lambda}(t+1), \hat{p}_b(t+1), \hat{T}_\chi(t+1)). \quad (15)$$

$T_{MAC}$ and $\hat{T}_{MAC}$ are depicted in Fig. 4 a) and b), respectively. The results were obtained from the simulation previously described (see Table I). The estimate envelope follows the real $T_{MAC}$ time. In the periods A and C, the parameter $\rho$ is equal to 0.137 and 0.008, respectively. In the other periods the network has more load ($\rho = 0.344$ and $\rho = 0.447$, for the periods B and D, respectively). Fig. 4 shows that $T_{MAC}$ changes with the network load, and that the $T_{MAC}$ estimate comes close to the real $T_{MAC}$ envelope.

| TABLE I |
| SIMULATION PARAMETERS USED IN THE RESULTS PRESENTED IN FIGS. 3, 4, 5 AND 6. |

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic data rate</td>
<td>1 Mbps</td>
</tr>
<tr>
<td>PHY data rate</td>
<td>1 Mbps</td>
</tr>
<tr>
<td>Data rate</td>
<td>1 Mbps</td>
</tr>
<tr>
<td>Simulated Time</td>
<td>1000 s</td>
</tr>
<tr>
<td>$p_b$</td>
<td>0.25</td>
</tr>
<tr>
<td>MAC Buffer Size (K)</td>
<td>200 frames</td>
</tr>
</tbody>
</table>
C. Adaptive $\beta$

A beacon inserted in the MAC transmission queue will wait a larger time when the buffer occupancy increases. The maximum waiting time occurs when the MAC buffer is full and can be approximated by $(K - 1) \max \{T_{MAC}(t + 1)\}$ ($K$ indicates the MAC queue size). A simple and brute force rule to adapt $\beta$ is to consider the maximum waiting time in the queue. The rule is described by \(^4\)

$$\beta = 1 + \text{ceil}\left[\left( K - 1 \right) \max \{T_{MAC}(t + 1)\} \right].$$

(16)

The MAC queue size $K$ can be a constant parameter for all nodes. Otherwise it can be transmitted in the beacon message. When larger MAC buffers are considered (bigger $K$ values), (16) can overvalue $\beta$. Although this overevaluation could penalize the virtual links accuracy when high mobility values are considered, it can also improve the BS accuracy because it can prevent unexpected MAC queue congestions during the future beacon period. This improvement assumes more relevance when the network is more loaded. For low network usages, the $T_{MAC}$ presents low values, which attenuates the $\beta$ overvaluation.

Fig. 5 presents the adaptive $\beta$ value for the simulation results presented in Figs. 3 and 4. $T_B = 1$ second was used. Comparing the periods C and D, the period D is more critical as the network is more loaded. For the period C, $\beta$ is most of the time at its lowest possible value of 2 (as stated by (16)). In the period A (from 0 to 250 seconds), the network traffic is bigger than in the period C. Thus $\beta$ is almost of the interval at 3, but it changes from 2 to 6. In the period B, $\beta$ changes from 3 to 13. The value 13 is achieved around 460 to 470 seconds when $T_{MAC}$ is maximum (see Fig. 4). Observing the Fig. 4, $T_{MAC}$ also has a bigger value during these instants. The same follows for the period D, where the traffic adapts $\beta$ from 4 to 25.

Fig. 6 presents the virtual links duration cumulative distribution function (CDF) when: only beacons messages are exchanged (without traffic) using a constant beaconing timeout of $T_O = 2.5T_B$; beacons and traffic are exchanged using a constant beaconing timeout of $T_O = 2.5T_B$; beacons and traffic are exchanged using the adaptive $\beta$. As the network is fully connected during the simulation time (1000 seconds), all the physical links last the 1000 seconds of simulation. The results show that the constant timeout should be avoided. Moreover, using the adaptive $\beta$ the BS virtual links duration match the physical links.

IV. PERFORMANCE RESULTS

When mobility is not considered, as in the results presented in Fig. 6, a high and constant $\beta$ value leads to accurate results (by absurdity it can converge to infinite). When mobility is assumed, a large $\beta$ value does not improve the BS accuracy. Now if a physical link breaks, its correspondent virtual link only is considered broken a time $T_O$ after. This section...
presents an analysis of the adaptive $\beta$ considering distinct node’s mobility patterns, in order to quantify if the computed adaptive $\beta$ values erroneously extend the virtual links duration. The random waypoint model was used to define two mobility scenarios. 10 nodes with 100 meters of communication range were randomly positioned within an square with 150x150 meters. The model parametrization is presented in Table II. The beacon transmission rate depends on the network mobility. The IEEE 802.11b MAC protocol and the simulations were parameterized according to the parameters shown in Table I. To increase the frames collision probability, each node generates short frames (with a constant 125 bytes payload) accordingly to an exponentially distributed inter-frame time with an average rate of 50 frames/second. The estimation period was set to $T_{est} = T_B/10$.

<table>
<thead>
<tr>
<th>scenario</th>
<th>$v_{min}$ [m/s]</th>
<th>$v_{max}$ [m/s]</th>
<th>pause time [s]</th>
<th>av. speed [m/s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1.0</td>
<td>40.0</td>
<td>6.0</td>
<td>5.84</td>
</tr>
<tr>
<td>B</td>
<td>10.0</td>
<td>40.0</td>
<td>0.0</td>
<td>21.64</td>
</tr>
</tbody>
</table>

Fig. 7. $T_{MAC}$ Interpolation error ($n=10$, $p_B=0.25$, $\lambda = 50$).

To compute $T_{MAC}$ in real-time, a look-up table was generated containing the values of each entry ($T_{\chi,n,A,p_B}$). Fig. 7 presents an example of the error obtained by applying a linear interpolator to the look-up table. The table error never goes above 8%.

Fig. 8 shows the virtual links duration CDF for the mobility scenario A defined in Table II. $T_B$ was set to 1 second, according to [6] (the scenario uses the maximum node’s speed of 40 m/s). The results show that the adaptive $\beta$ performs better when compared to the constant $\beta = 2.5$ curve.

The "Without Traffic" CDF link duration curve is plotted considering absence of network traffic. This curve is a benchmark, as it is similar to the network’s physical links duration.

For the higher mobility scenario B, [6] also proposes to parameterize $T_B = 1$ second. Using $T_B = 1$, the adaptive $\beta$ extends the virtual links duration, as can be seen in Fig. 9. The virtual links obtained using the adaptive $\beta$ are now erroneously prolonged because the beacon period $T_B$ is too large. Although defined according to [6], the $T_B$ value proposed by [6] is only based on node’s mobility issues. But our simulation also includes the network traffic, which effectively delays the beacon transmission and is not considered by [6].

For the same scenario we decrease $T_B$ to 0.25 seconds. The CDF curve for $T_B=0.25$ is plotted in Fig. 9. Now the virtual links duration almost match the physical ones. The beacon period adopted is enough to guarantee that the neighborhood is refreshed at a sufficient rate in order to address the network’s mobility and traffic constraints.

The results show that, if the $T_B$ meets the network’s traffic and mobility constraints, the virtual links accuracy is improved when the adaptive $\beta$ is used.

V. CONCLUSIONS

This work presents an adaptive timeout value for a Beaconing Scheme using the IEEE 802.11 MAC protocol. An analysis is proposed to compute the adaptive beaconing timeout based on a IEEE 802.11 model, and assuming node’s homogeneous load. The simulation results show that a constant beaconing timeout should not be used, as the BS presents very low accuracy. The adaptive timeout improves the BS accuracy. The results obtained with the adaptive timeout are similar to the physical link duration. Thus, the adaptive beaconing timeout improves the BS accuracy without prolonging the virtual links duration when node’s mobility is considered.

Further work will include an evaluation of the adaptive timeout to end-to-end flow performance, measuring its impact.
in the routing protocol used. Minor optimizations in the BS will also be addressed (e.g. deduce node presence from unicast traffic omitting some beacon transmissions).

REFERENCES


