Simulating and Modeling Wireless MAC Protocols

Willi Menapace, Mat. number 199890
willi.menapace@studenti.unitn.it

I. Introduction

The work aims at a better understanding of contention based wireless MAC protocols. A basic Aloha-like wireless MAC simulator and a specific network topology are given. The objective is to extend the capabilities of the basic MAC protocol with more sophisticated techniques, and explain its performance in the given network topology using both simulations and analytical modeling.

The work starts with an introduction to the chosen MAC extension and explains the structure of the relative simulator. It then proceeds with the illustration of the main simulation results and terminates with an analytical modeling of the MAC protocol.

II. Basic Information

The MAC extension chosen for implementation is the one referred to as 'Carrier Sensing with Collision Avoidance' plus 'Realistic Propagation' in the assignment paper. In order to better understand the simulator architecture and the analysis results, this section gives an overview of the implemented protocol.

As the name suggests, the implemented protocol features carrier sensing and is provided with a collision avoidance feature implemented through a contention window with a fixed number W_c of slots of duration T_s. When a node needs to transmit a packet on the channel, it starts by sensing it for a listening time period $T_l = T_s$. If the channel is sensed free, the packet is transmitted on the channel. Otherwise, if the channel is sensed busy or if the current node was transmitting a previous packet immediately before, a contention window of size W_c is opened. A random number B_k in the interval $[0, W_c - 1]$ is chosen. This number represents the chosen slot for transmission. At the end of every slot of duration T_s, B_k is decremented. If B_k reaches 0, the node starts the transmission in that slot, while if the transmission of another node is sensed during the countdown, the countdown is aborted and at the end of the other transmission the contention procedure restarts with a new extraction of B_k. Note that after the end of a transmission the contention procedure is started without listening to the channel first.

At least three major differences with 802.11 can be noted, namely the fixed contention window dimension, the extraction of a new value for B_k whenever the contention window is opened and the lack of listening before opening a contention window after a transmission.

The realistic propagation feature consists in a simulation of packet error rates during transmission. Specifically, when a packet is transmitted on the channel, it has a probability of corruption which increases linearly with the distance between sender and receiver nodes. In case of corruption, the destination node is able to sense the presence of a packet on the channel, but is not able to correctly receive it. Note that it is a very simplistic approximation of packet error rates, both because of the linearity assumption and the independence of the corruption probability from the length of the packet.

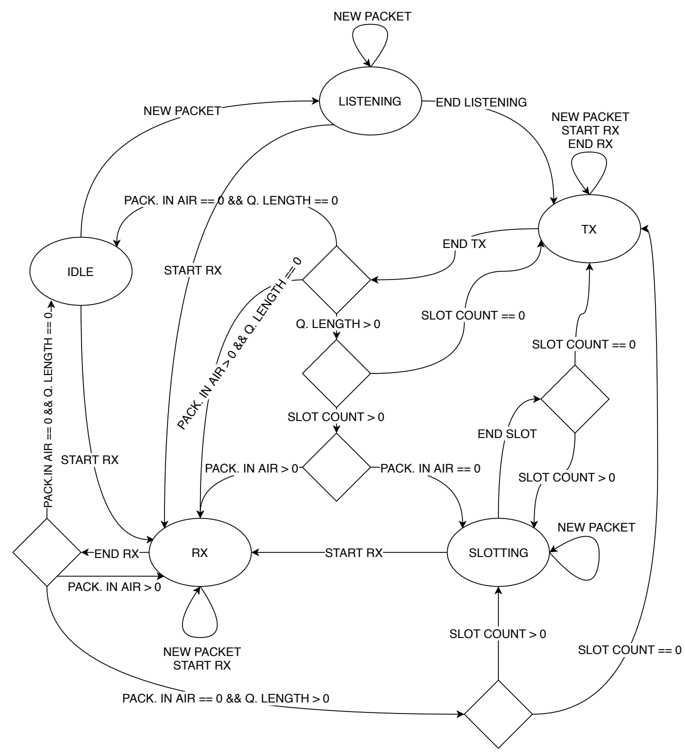


Figure 1. State machine associated to each node in the simulator for the proposed MAC protocol. Arrows departing from a state are labeled with the corresponding event. Diamonds represent choices, and leaving arcs are labeled with the correspondent condition. Some states do not have an associated arrow for every event. These events represent error conditions and have been omitted to avoid visual clutter.

III. Simulator Architecture

In order to analyze the performance of the proposed MAC protocol, it has been implemented in a discrete time event simulator making many modifications to the given Aloha-like one. The state machine associated to each node in the simulator is shown in Figure 1.

The simulator is build upon 5 states, 6 event types and many state variables.

Some of the most important state variables are the following:

queue length
The number of packets in the transmission queue including the packet in transmission

slot count
Corresponds to the variable B_k and indicates the number of slots to wait during contention

packets in air
The number of packets which are overlapping on the channel at the moment. Used to keep track whether the channel is free or not

The event types are the following

start rx
A new packet appears on the channel

end rx
The end of a packet on the channel is reached

end tx
End of transmission of the current packet in transmission

new packet
The node upper network layers request the transmission of a packet

end slot
End of a slot in the contention window

end listening
End of the listening period that precedes the transmission of a packet

The states are the following

idle
The node is in the idle state when its queue is empty and no packets are present on the channel. The only events that can originate are 'new packet' and 'start rx' which determine a change of state.

listening
The node, which was previously idle, is now sensing the channel to check if transmission of the packet arrived in the queue is possible. If the end of the listening period is reached then transmission starts, otherwise the node will switch to the rx state until the channel becomes free again.

tx
The node is transmitting a packet. In case a packet arrives on the channel, the node does not notice it and continues transmission. When the transmission ends four possible situations may happen. There are no packets to transmit and the channel is free, in which case the node goes to idle. There are no packets to transmit and the channel is busy, so the node notices energy on the channel and switches to the reception state. In case there are packets in the queue instead, a contention window is opened immediately without listening to the channel, differently from 802.11. If the assigned slot is the first, transmission happens unconditionally even if there are other packets on the channel due to collisions. Otherwise, the node can sense the channel. If there are packets on the channel the node aborts contention and switches to rx mode, otherwise if the channel is free, the node enters in the 'slotting' state where it waits for its assigned slot to begin.

slotting
A contention window is open and the node is waiting for the beginning of its slot to start transmission. Whenever an 'end slot' event occurs, the 'slot count' variable is decremented and the node starts transmission if it reaches 0. If a packet appears on the channel then contention is aborted and the node switches to the rx state.

rx
The node is receiving packets from the channel. Whenever an 'end rx' or 'start rx' event occur the variable keeping track of the packets on the channel is updated. The node is allowed to leave the reception state only when it senses the channel free which corresponds to the packets in air variable reaching 0. At this point, the node switches to state idle if there are no packets in the queue, otherwise it opens a contention window. Note that this behavior makes nodes vulnerable to channel capture.

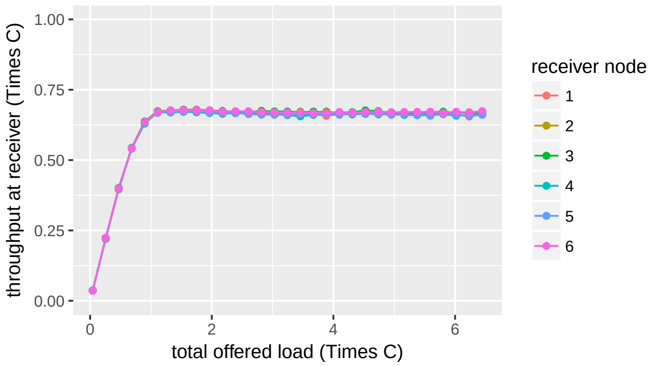
Many additional parameters have been embedded in the simulator. Some of them include the physical layer datarate which was kept to the original value of 8 Mbps, transmission range which was kept to $10\,\mathrm{m}$, $T_s = T_l$ which were set to $10\,\mu\mathrm{s}$, the transmission period of 10 bytes, W_c which was assigned values in the range from 1 to 1024 and the size of the transmission queue which was kept to 2 packets in order reduce transient periods to a minimum.

Two versions of the simulator have been produced, one without the 'realistic propagation' option and another with also this extension implemented. The first version was used as a reference point in order to easily understand the effects of propagation errors on the transmission. The code for this extension is embedded in the module that handles the behavior of the channel. Both versions were used in sanity tests to ensure their soundness and correctness.

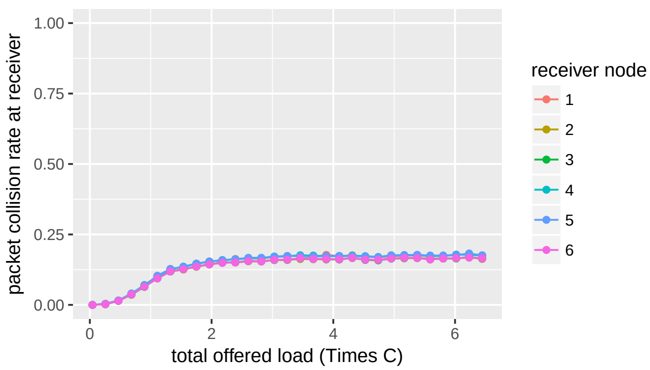
Before simulating the assigned network topology, a study on simpler topologies without realistic propagation was performed with the aim of gaining insights into the behavior of the protocol.

IV. Simulation of Single Collision Domain

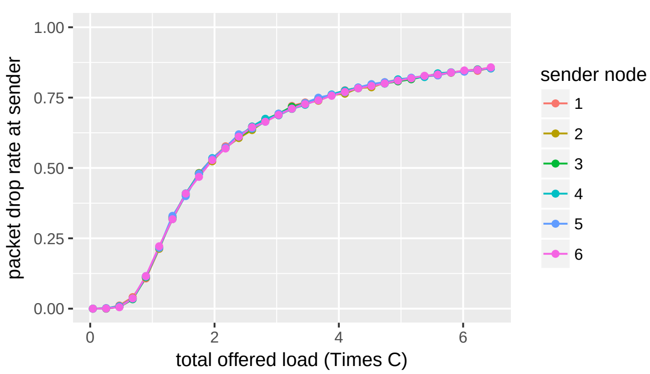
The first simulated topology regarded the simplest case in which every node was in range of every other node. This represents an ideal case because it does not suffer from phenomenons such as hidden terminals. A variety of situations have been tested varying the number of nodes and W_c. Figure 2 shows the results for 6 nodes and a contention window size of 32 slots, while Figure 3 represents the same situation with a contention window size of only 1 slot, that is, a protocol with simple carrier sensing without a collision avoidance feature. In order to be able to explore a wide number of configurations, only one seed has been used for each simulation.



(a) throughput

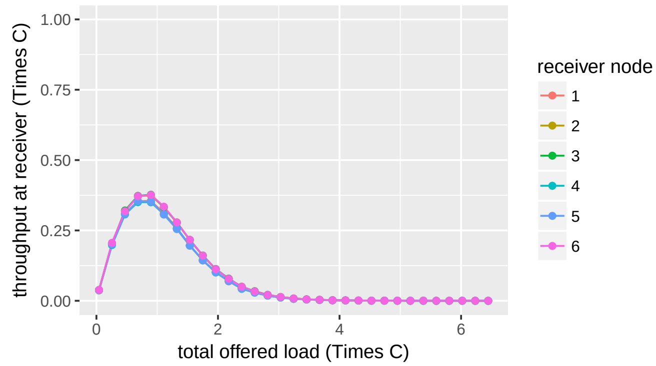


(b) collision rate

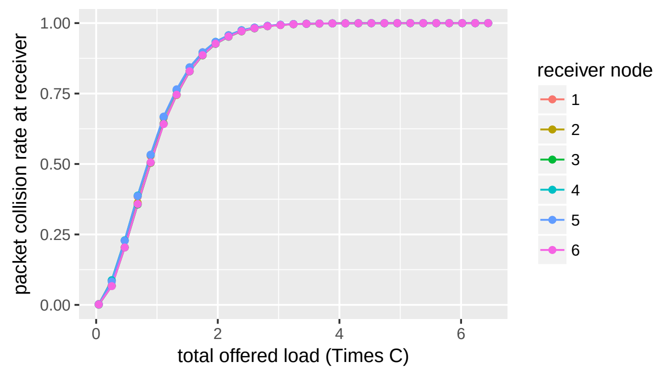


(c) drop rate

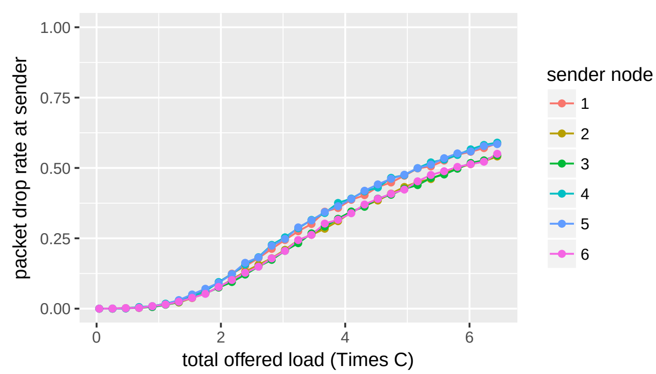
Figure 2. Simulated throughput, collision rates and drop rates of 6 nodes in mutual range with a collision window size of 32 slots. Offered load is normalized with respect to C, the channel capacity.



(a) throughput



(b) collision rate



(c) drop rate

Figure 3. Simulated throughput, collision rates and drop rates of 6 nodes in mutual range with a collision window size of 1 slot, situation representing the lack of a contention window and thus a simple carrier sensing protocol without collision avoidance feature. Offered load is normalized with respect to C, the channel capacity.

In the first case, the behavior of the protocol is almost ideal. The throughput grows linearly with the offered load until the load reaches $1C$, the capacity of the channel. At that point the channel reaches saturation and the throughput remains stable as the offered load grows. The packet collision rate remains low during the initial linear growth phase, and raises during the transition from linear growth to channel saturation, after which it stabilizes to a constant value.

The case with $W_c = 1$, which represents simple carrier sensing, is also very interesting. The obtained throughput closely resembles the one for the Slotted Aloha protocol, which features a maximum throughput of $0.37C$ when the offered load is $1C$. This is not a case. If we remember the classic Slotted Aloha modeling[1], it is found that the probability of collision of a packet is the probability that another station on the channel transmits in the same time slot where the slots are established a priori. In our simple carrier sensing protocol the situation is the same: the probability of collision of a packet is the probability that another station on the channel transmits in

[1] https://en.wikipedia.org/wiki/ALOHAnet

the same time slot where the beginning of a slot is given by the end of the previous packet and is not established a priori.

The analysis of other situations also highlighted that the optimal window size depends on the number of stations in the domain. In order to obtain the best throughput, it is better to increase the size of the window as the number of nodes grows. This corresponds to lowering the persistence of the protocol as the number of stations grows in order to avoid an increase in the collision rates. This is one of the reasons why more sophisticated protocols such as 802.11 feature a variable window size which enlarges at each collision in order to automatically reduce the persistence of the protocol and increase performance.

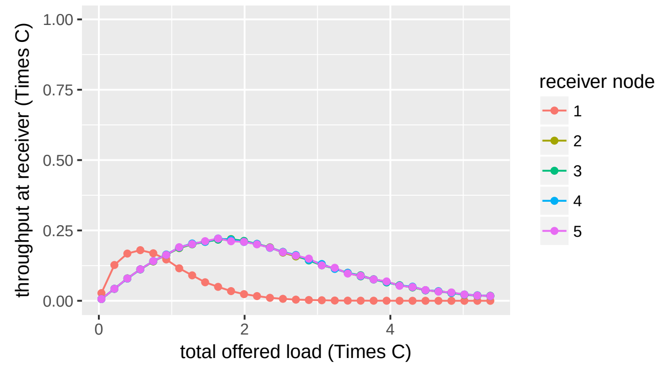
It is also possible to build an analytical model to predict the saturation throughput based on the window size and the number of nodes. This will be explored in Section VII.

V. Simulation of Hidden Terminal topology

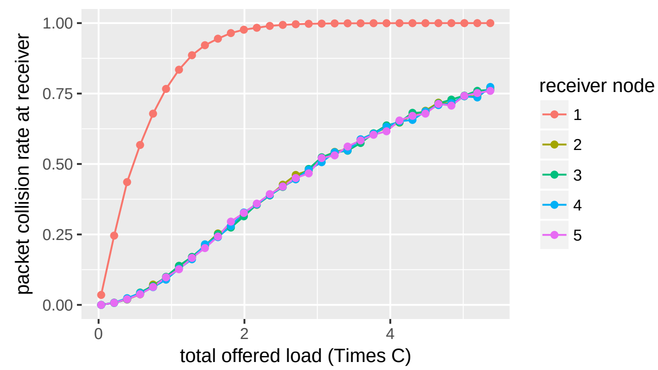
The second simulation regarded a topology subject to the hidden terminal problem. The topology is the one depicted in Figure 6, where a central node receives traffic from other 4 nodes which are not in mutual range. Many variations have been tested changing the size of the contention window in a range from 1 to 128 slots. Only one seed has been used in order to allow for a quicker analysis.

Figure 4 shows the results obtained for a collision window size of 1 slot, while Figure 5 show the situation in case of 128 slots. In both cases, the throughput obtained at the central node closely resembles the throughput curve typical of Pure Aloha which features a peak throughput of $0.18C$ at a load of $0.5C$. The result is fairly natural, in fact, being nodes 2-5 out of range, no coordination can happen between them and the carrier sensing and collision avoidance features are ineffective. New frames arrive at node 1 independently one another, even when the channel is occupied by another frame which is the typical Pure Aloha behavior.

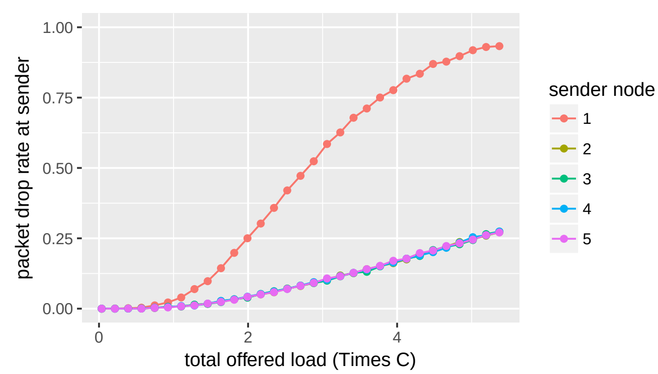
More interesting results are given by the behavior of the throughput curve at nodes 2-5 with different window sizes. These nodes can receive only traffic originated by node 1. A window of size 1 maximizes the throughput at those nodes because if node 1 has a packet in the queue it can be transmitted every time the end of a packet on the channel is reached. This is a consequence of our protocol implementation choice not to forcefully listen to the channel if a contention window is opened at the end of a reception. The high collision rates at nodes 2-5 confirm the result. On the other hand, with a higher window size, the throughput at nodes 2-5 follows the same curve for small channel loads and then drops significantly faster. The reason for this behavior is that when the channel load increases, the channel tends to be always occupied at node 1. While a contention window of size 1 allowed node 1 to always transmit anyway, with a large contention window node 1 will be forced to choose a slot which probably won't be the first one. In this case node 1 will sense the channel busy in the first slot and will abort contention without transmitting. This is an instance of channel capture which is confirmed by the collision rates at nodes 2-5, which remain close to 0, indicating that node 1 does not transmit often at higher loads. A way to improve performance would be not to reset B_k, the slot counter, each time the contention starts, giving node 1 a deterministic chance to transmit even in such situations. This strategy, in fact, is employed in 802.11 that also employs more sophisticated systems such as RTS-CTS mechanisms to mitigate the hidden terminal problem.



(a) throughput

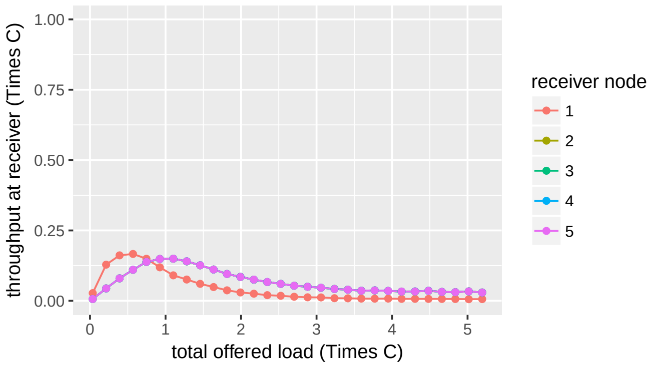


(b) collision rate

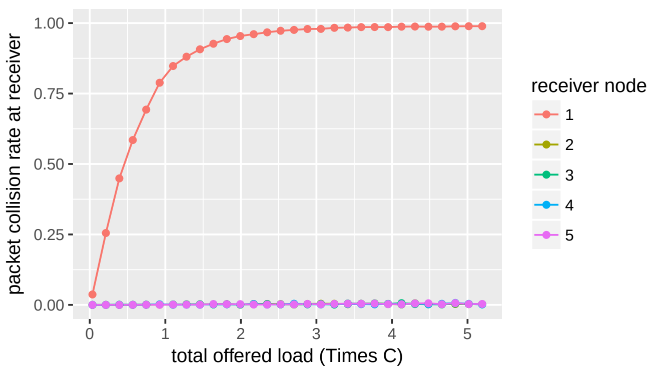


(c) drop rate

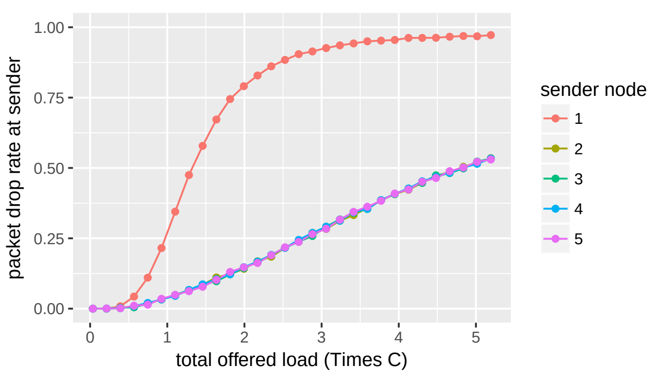
Figure 4. Simulated throughput, collision rates and drop rates in the hidden terminal topology with a collision window size of 1 slot. Offered load is normalized with respect to C, the channel capacity.



(a) throughput



(b) collision rate



(c) drop rate

Figure 5. Simulated throughput, collision rates and drop rates in the hidden terminal topology with a collision window size of 128 slots. Offered load is normalized with respect to C, the channel capacity. Lack of collisions on nodes 2-5 indicate that node 1 is blocked by channel capture.

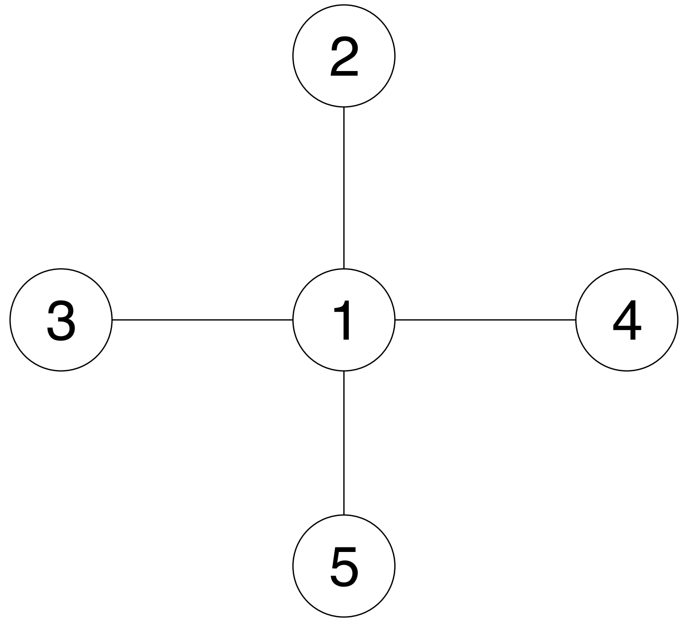


Figure 6. The hidden terminal topology used for the simulation. Lines connect the nodes which are in range.

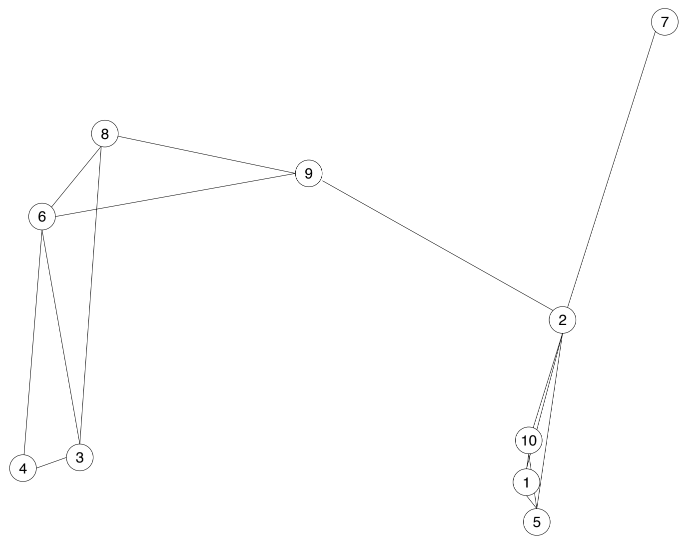


Figure 7. The complete assignment topology. Lines connect the nodes which are in range. Notice that many nodes are in range but very far from each other.

VI. Simulation of the complete topology

With the insights gained through the simulation of the simpler cases, the complete topology, depicted in Figure 7, was simulated. The topology was simulated with window sizes in the range from 2 to 128 slots, both with and without the realistic propagation extension. In order to obtain statistically sound results, each run was repeated with 21 different seeds and the plots in this section show the averaged results. It would be possible to obtain a confidence region for each plotted curve, but each run showed only very little variance, so this passage was not performed.

With the purpose of obtaining clearer plots and to simplify the discussion, the nodes were divided in different groups. Moreover, as the effects of varying window sizes are discussed in detail in the previous section and would not add new reflection possibilities to the explanation, only the case with a window size of 32 slots is discussed.

The first group of nodes to be analyzed is the cluster made by nodes 1, 5 and 10. Its throughput is shown in Figure 8.

The nodes are all in range and are not subject to the hidden terminal problem. The situation is ideal and the analysis performed in Section IV fully applies. Notice that the realistic

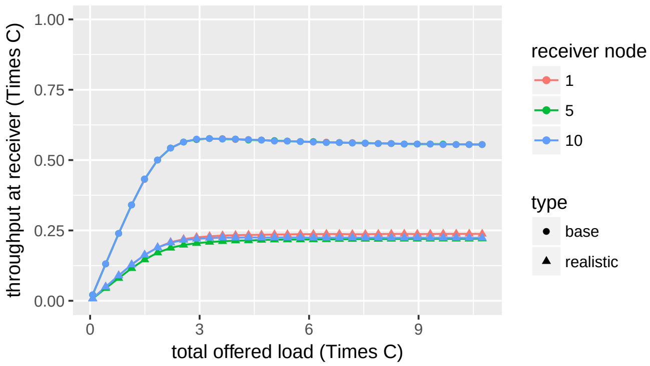


Figure 8. Simulated throughput of nodes 1, 5 and 10 with both simple and realistic propagation. Offered load is normalized with respect to C, the channel capacity.

propagation extension caused a constant reduction of the throughput that is to be attributed to the corruption of packets due to the distance traveled.

The second group to be analyzed is made of nodes 2 and 6. These nodes receive traffic coming from other nodes non in mutual range, so are subject to the hidden terminal problem present the behavior described in Section V. The results are shown in Figure 9.

The throughput curves follow a behavior similar to Pure Aloha. Node 2 reaches its maximum throughput when its 5 neighbors generate a traffic of $0.5C$, i.e. the global load offered by the 10 stations is $1C$. Node 6 reaches maximum throughput when its 4 neighbors offer a load of $0.5C$ which translates in a global offered load of $\frac{5}{4}C$. In general, the optimal global offered load in those cases can be obtained by solving the following equation.

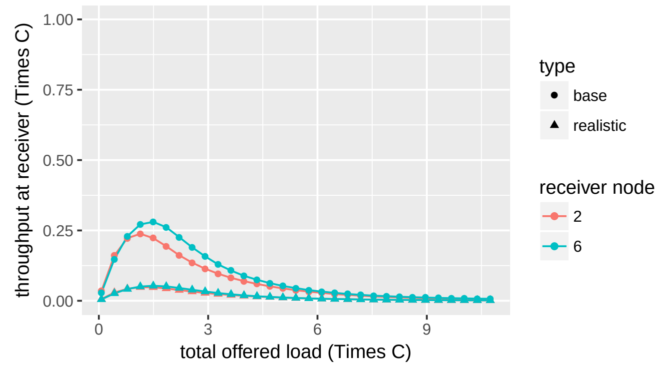
$$\frac{\text{number of neighbors}}{\text{total number of nodes}} * x = 0.5C$$

In case of realistic propagation the throughput is reduced by a constant factor due to the number of packets that get corrupted by the channel. The portion of packets corrupted by the channel can be seen in Figure 9b and is represented by the difference between the realistic propagation curves and the basic propagation ones.

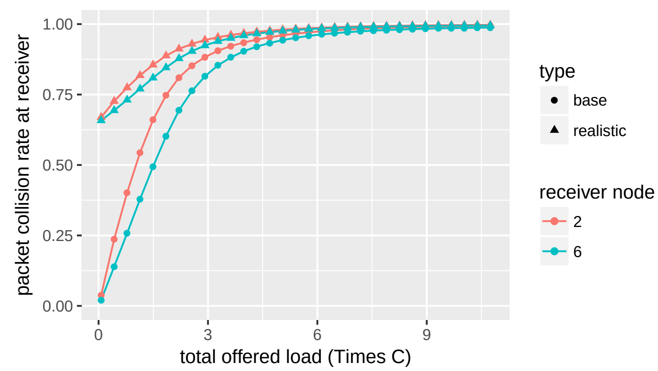
Both nodes are subject to channel capture and stop transmitting when the offered load increases.

The analysis then proceeded to nodes 3 and 8. These nodes have almost identical behavior. They share 2 of their 3 neighbors and the other neighbor is connected in the same way. The results are shown in Figure 10.

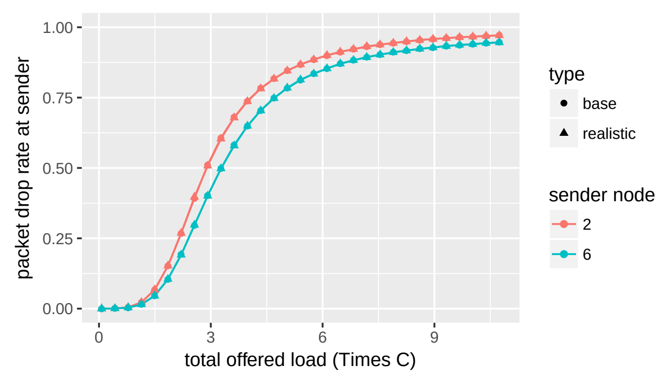
The nodes are in a nearly optimal situation as the one described in Section IV, with the only difference that nodes 8 and 4 (and symmetrically 3 and 9) are not in range and thus tend to generate collisions at node 3 (and respectively 8), lowering the throughput. The high number of collisions lowering the throughput from the ideal one is confirmed by Figure 10b. As usual, the realistic propagation version features a throughput reduced by a constant factor and the number of



(a) throughput



(b) collision rate



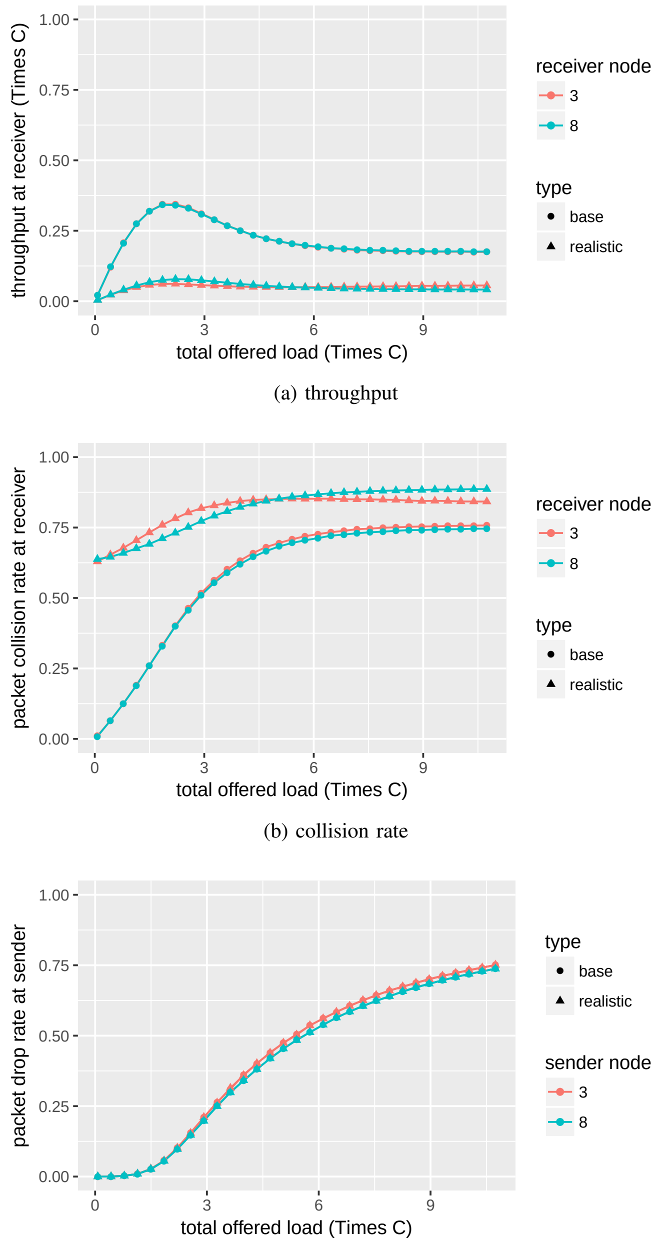
(c) drop rate

Figure 9. Simulated throughput, collision rates and drop rates of nodes 2 and 6 with both simple and realistic propagation. Offered load is normalized with respect to C, the channel capacity.

corrupted packets can be see from the packet collision rate plot.

Nodes 4 and 9 also have symmetrical behavior, with the only difference that node 9 is connected to node number 2, whereas node 4 does not have a similar connection. Their behavior is close to the ideal one described in Section IV because all the neighboring nodes, with the exception of node 2, are in mutual range. Results are shown in Figure 11.

The throughput at node 4 is close to the ideal one, while the one at node 9 is lower but eventually reaches the same value of node 4 with high channel loads. The reason is the collisions

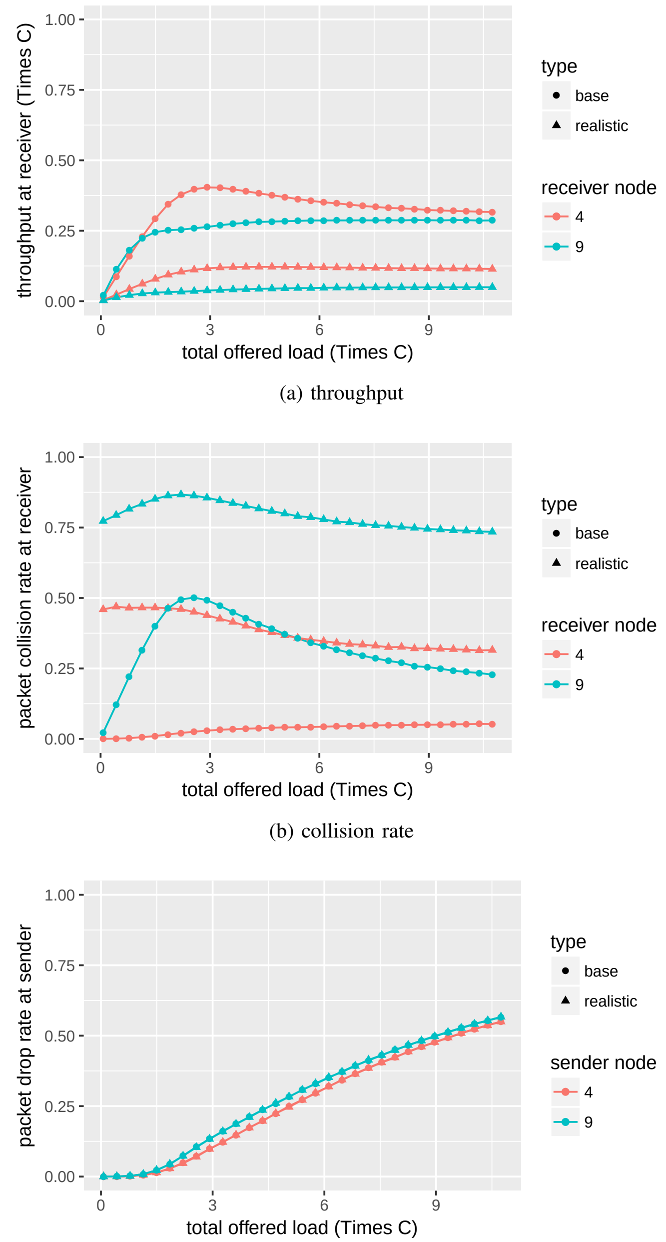


(a) throughput

(b) collision rate

(c) drop rate

Figure 10. Simulated throughput, collision rates and drop rates of nodes 3 and 8 with both simple and realistic propagation. Offered load is normalized with respect to C, the channel capacity.



(a) throughput

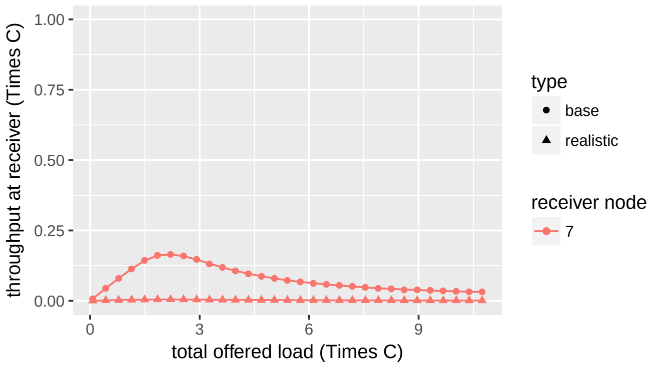
(b) collision rate

(c) drop rate

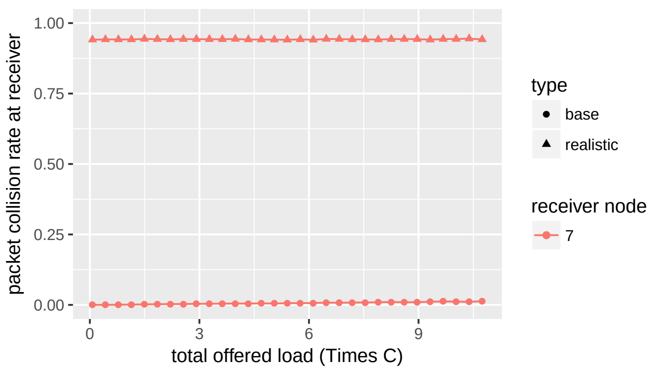
Figure 11. Simulated throughput, collision rates and drop rates of nodes 4 and 9 with both simple and realistic propagation. Offered load is normalized with respect to C, the channel capacity.

generated by the presence of node 2 which can be clearly seen in Figure 11b. When the offered load increases, however, node 2 is subject to channel capture and stops transmitting, thus reducing the collisions at node 9 and letting it behave more closely to node 4. Another interesting behavior lies in the packet collision rate curve of node 4 in case of realistic propagation where the number of collisions decrease as the load increases. The reason is that, as the load increases, node 6, which is the farthest one, is subject to channel capture and reduces its transmission rate progressively. The other neighbor transmitting is node 3 which is very close and causes a reduced number of packets corrupted due to the distance.

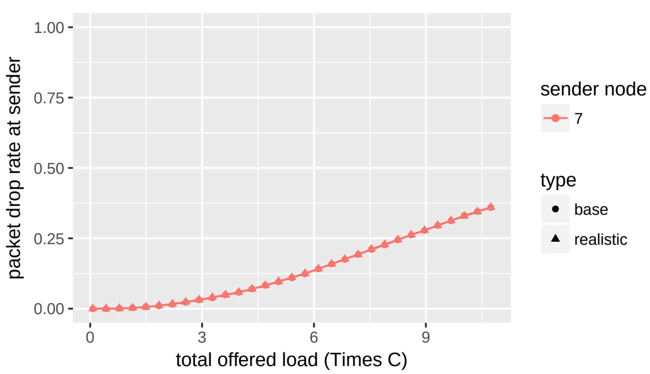
The last node to be analyzed is node 7, which results are shown in Figure 12. The node is at the borderline of node 2's range and should feature a behavior that is ideal because of the lack of hidden terminals. However, due to the channel capture of node 2, the throughput tends to 0 in case of the basic propagation. In case of realistic propagation instead, a high percentage of packets gets corrupted by the channel as shown in Figure 12b and thus the throughput always remains close to 0.



(a) throughput



(b) collision rate



(c) drop rate

Figure 12. Simulated throughput, collision rates and drop rates of node 7 with both simple and realistic propagation. Offered load is normalized with respect to C, the channel capacity.

VII. Analytical Modeling

This section tries to model analytically the behavior of the implemented protocol with a particular focus on the effects of the W_c parameter, the number of slots in the contention window, in order to find an ideal window value as a function of the number of nodes. The work finds its main inspiration in [1] and constitutes its adaptation to the proposed protocol. Both standard probabilistic techniques and Markov chains have been used.

The model aims at finding a closed form formula for the channel throughput in saturation conditions, that is, the throughput seen by the channel when the offered load is enough to ensure that every station always has at least one packet to transmit. In order to obtain such result, many assumptions have been made

- There are n stations in mutual range, each of which always has a packet to transmit
- The packets have fixed size S
- The counter B_k is not reset if the contention is aborted
- Propagation delays are negligible

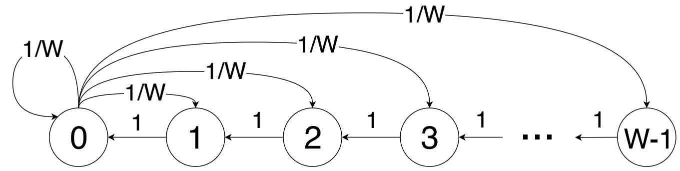


Figure 13. DTMC associated to the $B_k(t)$ process.

First of all, let $B_k(t)$ be the discrete time stochastic process that describes the change in time of variable B_k, the slot counter. The time instants t_i on which the variable B_k is sampled are the instants in which it changes value, so the time separating two sampling times can be either a slot time T_s or the time for the transmission of a single packet of fixed size S. According to the protocol, B_k can take values ranging from 0 to $W_c - 1$. The DTMC associated to the process is shown in Figure 13. The assumption that the counter B_k is not reset if the contention is aborted greatly simplifies the analysis of the chain. Without this assumption the chain would present arcs from each state to state $W - 1$ with an associated probability that depends on the steady state solution of the chain itself.

In order to find a steady state solution to the chain the following equations can be written

$$p_0 = \frac{1}{W}p_0 + p_1$$

$$p_1 = \frac{1}{W}p_0 + p_2$$

$$p_2 = \frac{1}{W}p_0 + p_3$$

$$\dots$$

$$p_{W-1} = \frac{1}{W}p_0$$

From which

$$p_i = (1 - \frac{i}{W})p_0$$

It is then possible to find p_0 posing the condition

$$\sum_{0}^{W-1} p_i = 1$$

Which yields

$$p_0 = \frac{2}{W+1}$$

The probability τ that the current node transmits a packet in the time period between two samplings of the process $B_k(t)$, which will be called frame, is given by p_0, the probability that in that frame the slot counter reached 0. As [1] suggests it is now possible to calculate the saturation throughput T as

$$T = \frac{E[\text{Transmitted bits per frame}]}{E[\text{Frame length}]} \tag{1}$$

where

$$E[\text{Transmitted bits per frame}] = n\tau(1-\tau)^{n-1}S \tag{2}$$

$$E[\text{Frame length}] = (1-\tau)^n T_s + (1-(1-\tau)^n)\frac{S}{C} \tag{3}$$

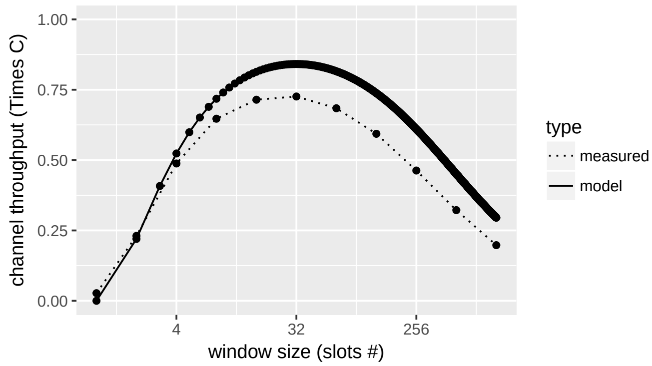
That is, the average successfully transmitted bits per frame is the packet size multiplied by the probability that exactly one node transmits, while the average frame length is the slot time in case no node is transmitting, and the time to transmit a packet if someone is transmitting. The assumption of fixed packet sizes allowed this last calculation to be simplified because each collided packet ends at the same time.

With the aim of validating the model, its predicted throughput curve was plotted against the curve obtained from simulation. The results are sound and are shown in Figure 14 for various numbers of stations. The model is able to capture fairly closely the saturation throughput in case that no hidden terminals are present. It predicts a progressive increase in the saturation throughput as the window size enlarges which reflects the lowering in the collision probability. After a certain point the throughput starts to fall as the window enlarges which represents the fact that the contention phase is becoming so long to determine an underutilization of the channel. The model tends to favor higher window sizes with respect to the simulation. This behavior might be attributed to subtle differences between the simulator and the model. First, the model assumes perfect saturation conditions, while in the simulator the nodes might not have packets to transmit in certain time instances reducing the need for higher window sizes. Another important difference is the behavior of counter B_k which is always reset in the simulator, but not in the model. Finally, the model assumes a fixed packet size, while the simulator features variable packet sizes.

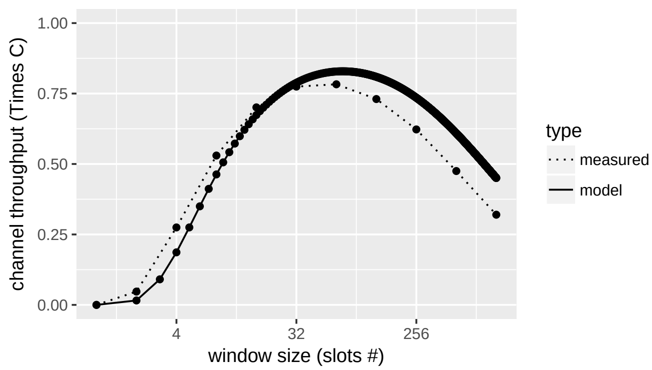
Notice that a higher number of transmitting stations requires a wider contention window in order to obtain the best results. This is a logical consequence of the fact that more transmitting stations cause more collisions and thus benefit from a less aggressive contention mechanism. This behavior is depicted in Figure 15 that shows the best window size as a function of the number of stations. The relation appears to be linear and a linear regression shows that the ideal window size can be described as

$$W_c = 13.2n - 8.48$$

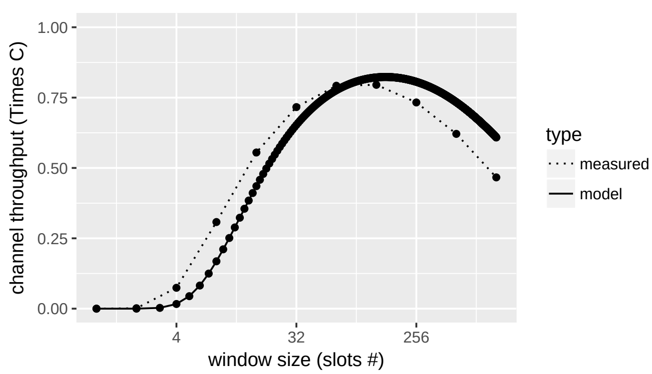
Notice that the proposed model is not able to capture the behavior of the throughput before saturation. Some attempts



(a) 3 nodes



(b) 6 nodes



(c) 12 nodes

Figure 14. Comparison of model saturation throughput against simulation saturation throughput with various numbers of stations in mutual range. The simulation throughput, which is relative to the throughput perceived by a single node and does not account for the data successfully transmitted by the measuring node, was multiplied by a factor $n/(n-1)$ in order to obtain the throughput as perceived by the channel. The window size axis is in logarithmic scale.

have been made to describe it without remarkable results. The best of these models is applicable only for high values of W_c where the number of collisions remains low. It consists in finding an upper limit T_{sat} for the throughput using the model described previously and then using the following formula.

$$T = \min(T_{sat}, \text{offered load})$$

The very rough assumption that this model makes is that

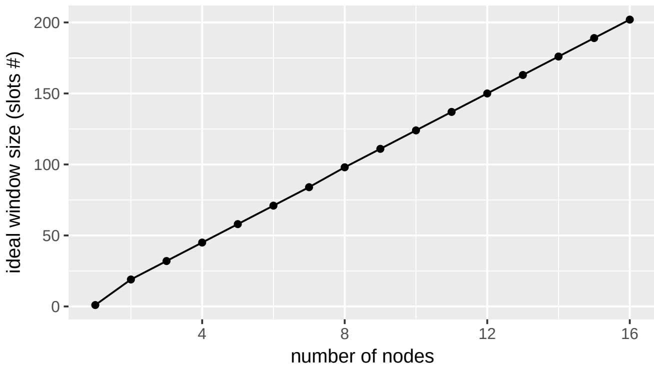
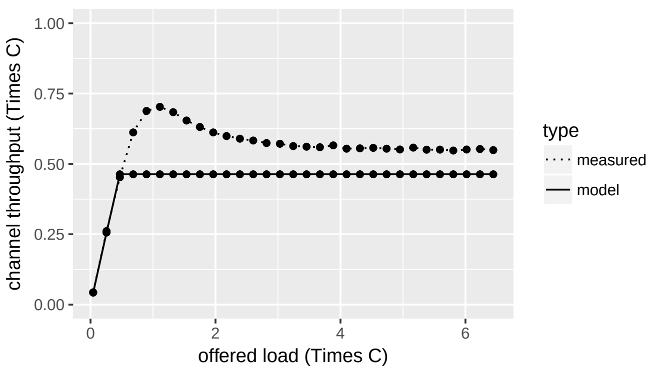
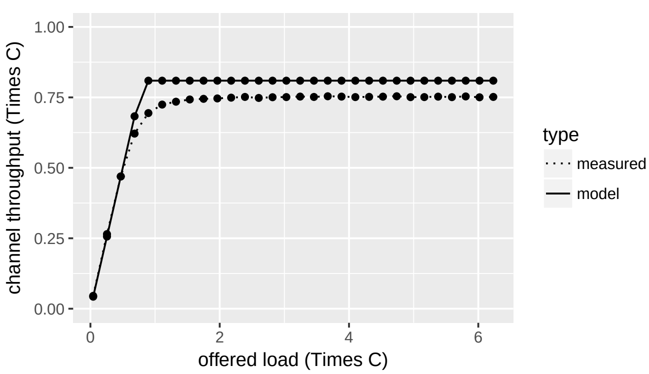


Figure 15. Ideal window size for optimal saturation throughput as a function of the number of transmitting nodes.



(a) $W_c = 8$



(b) $W_c = 128$

Figure 16. Comparison of model throughput curve against simulation throughput curve with different W_c in the case of 6 stations in mutual range. The model is not applicable in Figure 16a because the low collisions hypothesis is not respected while it provides good results in Figure 16b where the assumption holds. The simulation throughput, which is relative to the throughput perceived by a single node and does not account for the data successfully transmitted by the measuring node, was multiplied by a factor $n/(n-1)$ in order to obtain the throughput as perceived by the channel.

when the offered load is low, due to the wide contention window, collisions are absent and everything that is offered to the channel is correctly received until the limit established by T_{sat} is reached. A comparison of the model with simulation results is shown in Figure 16.

VIII. Conclusion

The work aimed at the analysis of a proposed network protocol on a given topology.

The simulation phase highlighted a good behavior of the protocol in the case that every node is in mutual range, showing stability characteristics and good usage of the available channel bandwidth. On the other hand, when the topology presented hidden terminals the performance of the protocol was comparable to that of Pure Aloha with channel capture phenomenons.

The subsequent analytical modeling phase was able to describe fairly accurately the behavior of the saturation throughput in relation to the number of stations and the size of the contention window in the case that every node is in mutual range. The model confirmed the intuition that a higher number of stations requires a wider contention window and provided a formula for calculating the ideal window size. A subsequent model provided a rough formula for calculating the throughput curve.

As a result, the author proposes some modifications taken from 802.11 that would enhance the performance of the protocol. First, the introduction of an RTS-CTS mechanism to mitigate the Pure Aloha behavior found in hidden terminal situations, which is not acceptable for high channel loads. Second, avoid resetting B_k whenever contention is restarted in order to reduce channel capture. Lastly, implement a variable contention window size in order to allow the protocol to dynamically adapt to the number of transmitting nodes on the channel.

IX. Appendices

A. Package Description

The software package is composed of the following files

- *.py files, the modified simulator files. In the current configuration the script performs simulation using the realistic propagation extension. In case that simple propagation is required, the user may enable it by commenting lines #113 and #114. The scripts can be started by running main.py with the original options or by using the convenience script worker.py which performs multiple simulation runs in parallel. worker.py allows to specify the number of threads to use for the computation by modifying line #13.
- config.json, contains parameters for the simulator. The current configuration is the one used for the simulation of the assignment topology. Running this configuration may require up to 5 hours of CPU time and 60GB of disk space.
- process.R, reads simulation data and produces both intermediate csv files, that can be used for further elaboration, and throughput, collision rate and drop rate plots. It does not require to read the whole dataset in RAM, differently from the original script. Dependencies are automatically installed if run with administrative privileges.
- plotting.R, reads intermediate csv files produced by process.R, which must be adequately renamed, and produces

plots for each specified group of nodes that compare simple propagation results with realistic propagation results. Dependencies are automatically installed if run with administrative privileges.

- model.evaluation.R, reads intermediate csv files associated with saturation throughput lambdas and compares simulation results with analytical model predictions. Dependencies are automatically installed if run with administrative privileges.
- simple.linear.model.R, reads intermediate csv files and compares simulation results with analytical model predictions for the throughput curve. Dependencies are automatically installed if run with administrative privileges.

References

[1] G. Bianchi, "Performance analysis of the ieee 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, March 2000.