Komondor: a Wireless Network Simulator for Next-Generation High-Density WLANs

Sergio Barrachina-Muñoz, Francesc Wilhelmi[*],
Ioannis Selinis, and Boris Bellalta

Abstract

Komondor is a wireless network simulator for next-generation wireless local area networks (WLANs). The simulator has been conceived as an accessible (ready-to-use) open source tool for research on wireless networks and academia. An important advantage of Komondor over other well-known wireless simulators lies in its high event processing rate, which is furnished by the simplification of the core operation. This allows outperforming the execution time of other simulators like ns-3, thus supporting large-scale scenarios with a huge number of nodes. In this paper, we provide insights into the Komondor simulator and overview its main features, development stages and use cases. The operation of Komondor is validated in a variety of scenarios against different tools: the ns-3 simulator and two analytical tools based on Continuous Time Markov Networks (CTMNs) and the Bianchi's DCF model. Results show that Komondor captures the IEEE 802.11 operation very similarly to ns-3. Finally, we discuss the potential of Komondor for simulating complex environments – even with machine learning support – in next-generation WLANs by easily developing new user-defined modules of code.

1 Introduction

The Institute of Electrical and Electronics Engineers (IEEE) 802.11 Wireless Local Area Networks (WLANs) are evolving fast to satisfy the new strict requirements in terms of data rate and user density. In particular, various IEEE 802.11 amendments have been introduced in the past few years or are under active development to accommodate the need for higher capacity, exponential growth in number of devices, and novel use-cases. [1]. An example of next-generation high-density deployment is depicted in Fig. 1 where multiple WLANs are allocated with different channels and dynamic channel bonding (DCB) policies.

Of particular interest is the IEEE 802.11ax (11ax) amendment [2], that is under active development and which was introduced to address the demands and challenges that WLANs will face in the congested 2.4/5 GHz bands [3].

[*] The contribution of the first two authors is the same.

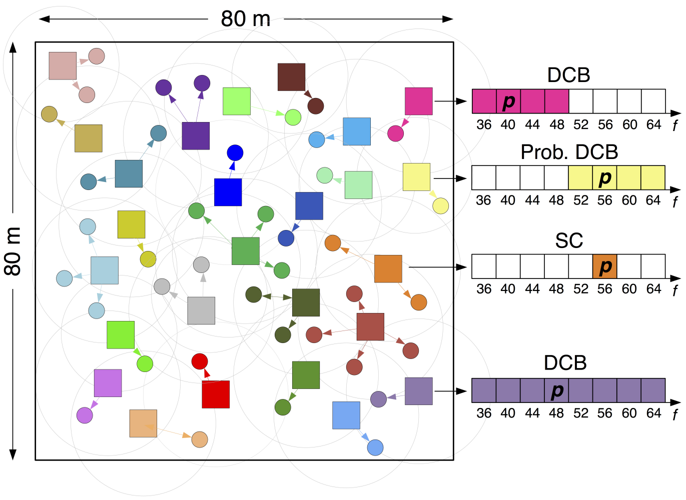

80 m
80 m
DCB
p
36 40 44 48 52 56 60 64 f
Prob. DCB
p
36 40 44 48 52 56 60 64 f
SC
p
36 40 44 48 52 56 60 64 f
DCB
p
36 40 44 48 52 56 60 64 f

Figure 1: Dense scenario composed of 25 WLANs. Note that each WLAN has its own channel allocation and DCB policy.

Other important amendments for next-generation wireless networks are the IEEE 802.11ay [4] and EXtreme Throughput (XT) 802.11 [5], which aim to exploit the 60 GHz and ≤ 6 GHz frequency bands, respectively. Amendments like the aforementioned ones lay the foundation of next-generation WLANs by including new features such as multiple-antenna techniques like Downlink/Uplink Multi-User Multiple-Input-Multiple-Output (DL/UL MU-MIMO), spatial reuse techniques like BSS coloring, and efficient use of channel resources like DL/UL Orthogonal Frequency Division Multiple Access (OFDMA). Therefore, it becomes necessary to provide reliable simulation tools able to assess the performance and behavior of next-generation WLANs in multiple scenarios/cases, especially in high-density deployments.

In this paper, we present Komondor,[1] an open source, event-driven simulator based on the CompC++ COST library [6]. Komondor is focused on fulfilling the need for assessing the novel features introduced in recent and future amendments, which may be endowed with applications driven by machine learning (ML). The motivation for developing and building the presented wireless network simulator is motivated by:

i) The lack of analytical models for capturing next-generation techniques in spatially distributed and/or high-density deployments.

ii) The lack of next-generation WLAN-oriented simulators.

[1] All of the source code of Komondor, under the GNU General Public License v3.0., is open, and potential contributors are encouraged to participate. The repository can be found at `https://github.com/wn-upf/Komondor`

iii) The complexity of extending simulators comprising an exhaustive implementation of the physical (PHY) layer.

iv) The large (or intractable) execution time required by other simulators to simulate high-density deployments.

v) The need for conveniently incorporating ML-based agents in the simulation tool.

In short, Komondor is designed to efficiently implement new functionalities by relying on flexible and simplified PHY layer dependencies, to be faster than most off-the-shelf simulators, and to provide reliable simulations and a gentle learning curve to new users.

2 Wireless network simulators

Wireless network simulators can be categorized into continuous-time and discrete-event. On the one hand, continuous-time simulators continuously keep track of the system dynamics by dividing the simulation time into very small periods of time. On the other hand, in discrete-event simulators, events are used to characterize changes in the system. Accordingly, for the latter, events are ordered in time and normally allow running faster simulations than continuous-time simulators. In addition, discrete-event simulators allow tracing events with higher precision.

From the family of discrete-event driven network simulators, only a few ones are publicly available. OMNET++ [7] is a component-based C++ simulation library that is not open-source and is used for modeling communication networks and distributed multiprocessor systems. OPNET is another commercial simulator that allows the integration of external components. NetSim [8] was conceived to provide an accurate simulation model oriented to the world wide web. To that purpose, the simulator was written in Java, which compromises simulation time with programming flexibility. When it comes to open source network simulators, a MATLAB-based link-level simulator was presented in [9] for supporting the IEEE 802.11g/n/ac/ah/af technologies. The ns-2 simulator [10] is another network simulator known for its accuracy and the integration with the network animator. Finally, the ns-3, which was introduced in 2006 to replace the ns-2, presents significant advantages over the ns-2 due to its detailed simulation features, becoming very popular among the research community [11]. Table 1 highlights in a nutshell the most important characteristics of the overviewed network simulators and Komondor.

Among the family of overviewed discrete-event simulators, we highlight the ns-3 open-source simulator due to its popularity and use it as a baseline for comparing against Komondor. Despite the plethora of features that are supported

[2] Although ns-2 and ns-3 do not provide a default graphical animation tool, there are tools supporting live animation, e.g., PyViz or NetAnim for ns-3 and NAM for ns-2.

[3] blueAn integration with OpenAI Gym has been recently provided to ns-3 [12], but the ML-based operation is not part of the simulator.

Table 1: Comparison of wireless network simulators.

Simulator	**Open-source**	**Source lang.**	**Complexity**	**GUI**	**11ax features**	**ML/based module**
ns-3	Yes	C++	High	No[1]	Partial	No[2]
ns-2	Yes	C++/OTcl	Low	No[1]	No	No
OMNET++	No	C/C++	Medium	Yes	No	No
OPNET	No	C++	Medium	Yes	No	No
NetSim	No	Java	Low	Yes	No	No
Komondor	**Yes**	**C/C++**	**Low**	**No**	**Partial**	**Yes**

in ns-3, it has some inherent limitations, such as the high complexity for developing new features/models as an extension of the simulator core. In particular, compatibility with the already existing/supported models is required and must be carefully ensured. For example, beamforming for previous mature amendments (i.e. IEEE 802.11n/ac) is not available yet, owing to the effort required to integrate it. Moreover, the integration of new features strongly depends on the willingness of the community to contribute to the development.

With respect to the IEEE 802.11ax operation – rates and support for information elements are being developed – the implementation is mostly based on the Draft 1.0 [13]. Such a draft dates from 2016 and does not include most of the core IEEE 802.11ax functionalities. At the time of submitting this paper, only the Single-User Protocol Data Unit (SU PPDU) and MIMO with up to four antennas are supported in ns-3, whereas OFDMA and MU-MIMO are not supported in the official distribution [14].

Apart from the official resources, we find few ns-3 works publicly available that support IEEE 802.11ax features, which may (or may not) be integrated into future releases. For example, we highlight the works with regard to the OFDMA that have been carried out by Getachew Redieteab et al. (based on the IEEE 802.11ax specification framework document [15]) and Cisco [16]. However, none of these works completely follow the latest developments in the IEEE 802.11ax standard and have not been validated through extensive simulations and testbed results, as had previously occurred with the OFDM [17]. In addition to OFDMA, the spatial reuse operation (i.e., BSS Color [18]) is under active development, whereas extensions of the capture effect have been applied to ns-3 to follow the IEEE 802.11ax guidelines and studied in [19] and in a testbed [20].

3 Komondor Design Principles

3.1 Architecture

Komondor aims to realistically capture the operation of WLANs. Henceforth, it reproduces actual transmissions on a per-packet basis. To that purpose, Komondor is based on the COST library, which allows building interactions between components (e.g., wireless nodes, buffers, packets) through synchronous and asynchronous events. While the former are messages explicitly exchanged between components through input/output ports, the latter are based on timers. In practice, components perform a set of operations until a significant event occurs. For instance, a node that is decreasing its backoff may freeze it when an overlapping node occupies the channel. The beginning and end of such a transmission are examples of significant events, whereas decreasing the backoff counter is not. Nevertheless, events may be triggered by different timers. In the previous example, a node's transmission begins once the backoff timer terminates (i.e., the backoff timer triggers the beginning of the transmission), while the end of the transmission is triggered by the packet transmission timer. Fig. 2 shows the schematic of a COST component, which is composed of inports, outports, and a set of timers.

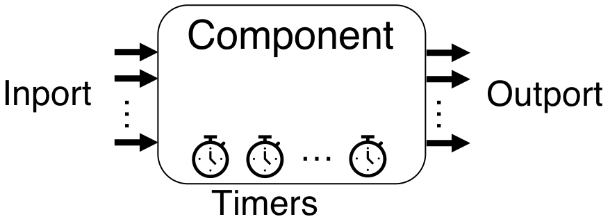


Figure 2: COST component. While inports and outports allow to directly communicate with other components, timers trigger events specific to the component.

3.2 IEEE 802.11 Features

Komondor entails a long-term project in which several contributors are involved. That is, the simulator is continuously evolving to include novel techniques and generally improve performance. The current version of Komondor (v2.0) includes the following fully tested IEEE 802.11ax features:

- **Distributed coordination function (DCF)**: the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) captures the basic Wi-Fi operation for accessing the channel. Moreover, Contention Window (CW) adaptation is considered.
- **Buffering and packet aggregation**: several traffic generator models are implemented in Komondor such as deterministic, Poisson or full-buffer. Besides, multiple media access control protocol data unit (MPDU) can be aggregated into the same PLCP protocol data unit (PPDU) in order to reduce the generated communication overheads.
- **Dynamic channel bonding (DCB)**: multiple channel widths can be selected during transmissions by implementing DCB policies in order to

maximize the spectrum efficiency. Some of these policies were already evaluated in [21], [22].

- **Modulation coding scheme (MCS) selection**: the MCS is negotiated between any transmitter-receiver pair according to the Signal-to-Interference-and-Noise Ratio (SINR), thus supporting multiple transmission rates.

- **Ready-to-send/Clear-to-send (RTS/CTS) and Network Allocation Vector (NAV)**: virtual carrier sensing is implemented in order to minimize the number of collisions by hidden-nodes.

Future development stages are under progress including other features such as OFDMA, MU-MIMO transmissions, beamforming, spatial reuse, and ML-based configuration.

3.3 Execution Flowchart

Komondor is composed of several modules that allow performing simulations with a high degree of freedom. Fig. 3 summarizes the operational mode of Komondor from a user's point of view. A more detailed user's guide providing a quick-start and guided execution examples is available in the Komondor's Github repository.

3.3.1 Input and Setup/Start

as for the execution console command for starting Komondor simulations, arguments are designed in a simple and efficient way. Examples of console arguments are the file names of the inputs, the activation flags of the logs, the simulation time and the random seed. In addition, input files (in CSV format) are used to define the environment and have been conceived in a way that the user can easily modify important simulation parameters such as the traffic load, the path-loss model, or the data packet size. Once the environment is generated and nodes are initialized, traffic is exchanged between nodes until the simulation time runs out.

3.3.2 Stop and Output

when the simulation finishes, the closing is handled and statistics are gathered. Then, extensive and detailed performance statistics are per default provided by Komondor (e.g., throughput, delay, spectrum utilization, or collisions). Moreover, the user can efficiently include as much as metrics as desired.

3.4 States and events

The Komondor's core operation is based on states, which represent the status (or situation) in which a node can be involved. A state diagram summarizing both states and transitions is shown in Fig. 4. Roughly, a given node starts

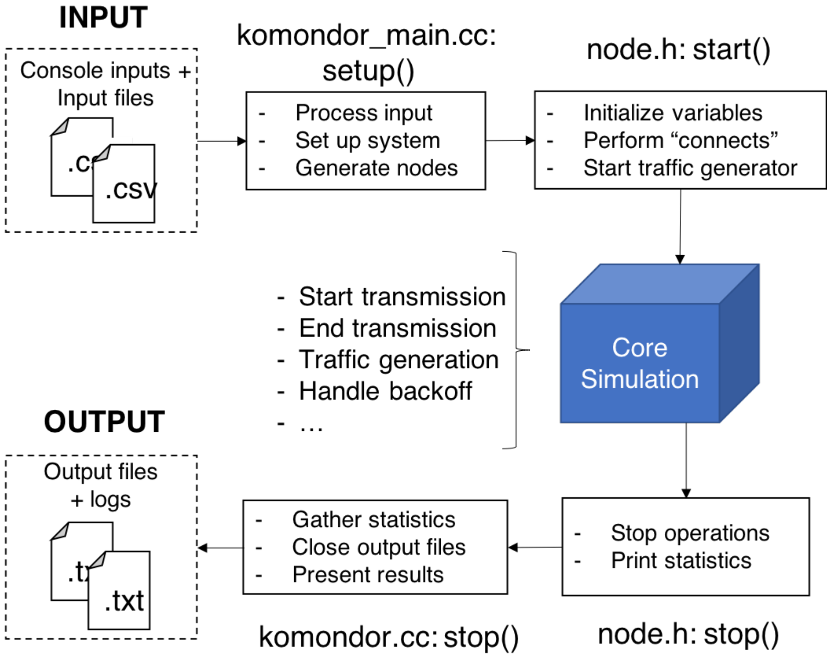


Figure 3: Komondor execution flowchart.

in the SENSING state, where multiple events can occur (e.g., a new packet is buffered or a new transmission is detected). Then, according to the noticed event, the node transits to the corresponding state.

3.4.1 States

we depict below each state and how a node must behave in front of new events.

- **SENSING**: a node senses the channel with two main purposes. First, to follow the CSMA/CA operation to gain access to the channel (in case there is backlogged data in the buffer/s). Second, to wait for incoming transmissions, so that either carrier sensing or receiving procedures are held. In case of being immersed in a backoff procedure, a node detecting a "new transmission" event would sense the power received in its primary channel, and assess whether to freeze the backoff countdown. Similarly, whenever an "end transmission" event occurs, the channel is sensed in order to determine whether the backoff counter can be resumed or remain paused.

- **TRANSMIT**: a transmitter node is currently transmitting a packet. No matter what events may occur, during the packet transmission, the node

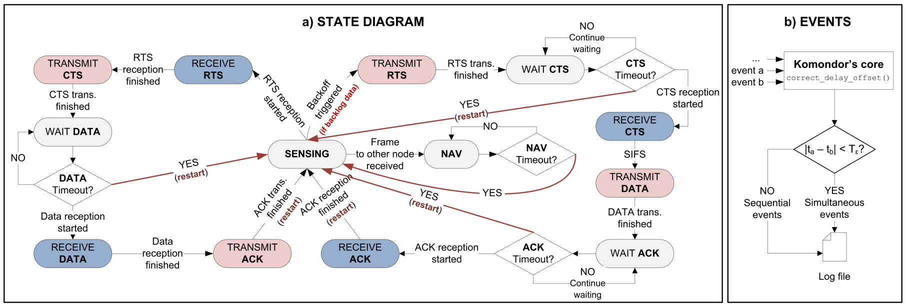


Figure 4: Komondor's state diagram and events. a) States are reachable by different transitions. b) Simultaneous events are properly processed through delay offset correction.

blocks its receiver capabilities and remains in the same state until the transmission is finished.

- **RECEIVE**: when a node is receiving and decoding an incoming packet, it will behave in front of a new event according to its implication in the channel of interest. Of especial importance are those new transmission events triggered by other nodes that have gained access to the medium. Specifically, if a new transmission generates enough interference, the ongoing reception will be discarded, thus leading to a packet loss.

- **WAIT states**: these states allow modeling the situations where a node that transmitted a packet is expecting for the corresponding response. Namely, after transmitting RTS, CTS or DATA packets, the transmitter will wait for the corresponding CTS, DATA or ACK/BACK packets, respectively. If the response packet is not received before the corresponding timeout is triggered, the transmitter assumes that either the transmitted packet or the response packet is lost and resets to SENSING state. Wait states are particularly useful to detect packet losses when anomalies in the network (e.g., hidden terminal problem) occur.

- **NAV**: when a node enters in NAV state due to the successful reception of a frame addressed to a different destination, it sets a NAV timer and keeps listening to its primary channel. If a new frame is successfully received during the NAV, the timer is updated, provided that the new NAV time is larger than the current remaining time.

3.4.2 Events

each time a node performs an action that can affect the system (e.g., it starts transmitting a frame), an event is announced. Events in Komondor are lined up on the time axis and handled by the core entity. Events management is similar in ns-3. However, the latter exhibits a significant limitation, since events that are scheduled at the exact same time can be executed in any order. Such a development feature may lead to unpredictable results and is incompatible with real-world situations in which events can occur simultaneously. Some inconsistencies may occur in case that the execution order affects multiple simultaneous events (e.g., two packets arriving at the exact same time). To solve this, Komondor, which is also a discrete-event simulator, employs temporal variables to compare the exact timestamps at which two or more events were generated. As a result, Komondor is able to successfully simulate the behavior of simultaneous events while keeping the logic of the states.

3.5 Developing new modules

Komondor has been conceived to be easily modified and extended. In particular, several modules have been provided to represent different simulation capabilities (e.g., propagation, channel access or traffic generation). Accordingly, Komondor can be potentially extended to support the operation of other IEEE 802.11 amendments such as 11n, 11ac, 11ad or 11ay. In addition, ML-based modules can also be introduced. A complete manual can be found at the Komondor's repository.

4 Validation

In this Section, we validate the operation of Komondor and show its potential for dealing with high-density scenarios. In particular, we show the reliability of the simulator, despite its reduced complexity of the PHY.[4] The validation of the Komondor's operation is done through a set of illustrative scenarios, and our results are compared with the ones obtained with ns-3.[5] In addition to ns-3, a mutual validation is performed with the Continuous Time Markov Networks (CTMNs) modeling introduced in [24], and which is extended for spatially distributed networks in the Spatial-Flexible Continuous Time Markov Network (SFCTMN) framework [21]. As for high-density scenarios, we make use of the Bianchi's DCF analytical model [25] to validate the results in fully-overlapping deployments, where all the nodes are within the carrier sense of the others. The results shown in the following subsections were obtained according to the parameters defined in Table 2. The duration of the RTS, CTS and data frame is computed as follows:

[4] For instance, channel effects are assumed to remain static during the whole transmission of a given frame, and the propagation delay is considered to be negligible.

[5] Details on the ns-3 implementation used in the simulations presented throughout this paper can be found at `https://github.com/wn-upf/Komondor/tree/master/Documentation/Validation/ns-3`. For instance, this implementation includes the 11ax residential scenario propagation loss [23] and has a PLCP training duration updated according to the 11ax amendment [2].

$$T_{\text{RTS}} = T_{\text{PHY-leg}} + \left\lceil \frac{L_{SF} + L_{\text{RTS}}}{L_{s,l}} \right\rceil \sigma_{\text{leg}},$$

$$T_{\text{CTS}} = T_{\text{PHY-leg}} + \left\lceil \frac{L_{SF} + L_{\text{CTS}}}{L_{s,l}} \right\rceil \sigma_{\text{leg}},$$

$$T_{\text{D}} = T_{\text{HE-SU}} + \left\lceil \frac{L_{\text{SF}} + L_{\text{MH}} + N_{\text{agg}} L_{\text{D}}}{L_{s,l}} \right\rceil \sigma.$$

Note that full-buffer traffic is assumed in all the scenarios throughout this work for comparative purposes. Moreover, we have considered the residential path-loss model recommended in the IEEE 802.11ax [23], which inflicts high losses due to its large number of obstacles (e.g., walls).

4.1 Analyzing toy Scenarios

Komondor has been conceived as a friendly and ready-to-use wireless network simulator that can be used by researches and teachers to study fundamental networking issues. In particular, scenarios and environment configurations can be conveniently modified through structured input files. The scenarios proposed in this Section are a clear example of toy scenarios where different networking concepts such as *flow starvation* or *additive interference* take place. Furthermore, a given user can easily analyze WLAN scenarios through the implemented logs generation system and statistics reporting. Accordingly, particular phenomena in the PHY and medium access control (MAC) layers can be tracked (e.g., channel contention, packet collisions, physical carrier sensing, energy detection, or buffer dynamics).

4.2 Basic Operation

We first aim to validate the basic IEEE 802.11 operation of the DCF implemented in Komondor when RTS/CTS is applied. For that, we consider a single Access Point (AP) scenario (we name it *Scenario 1*) with one and two stations (STAs), where full-buffer downlink traffic is held. The two-STAs case allows us to assess the proper behavior of Komondor in presence of multiple STAs. To validate this scenario, we compare the Komondor results with the ones provided by ns-3 and the SFCTMN framework. Fig. 5 shows the simulation results obtained from each tool, for packet aggregation ($N_{\text{agg}} = 64$) and no-aggregation ($N_{\text{agg}} = 1$). We note that the average throughput obtained by each simulation tool is almost identical, either for packet aggregation or not. In addition, having multiple STAs leads to the same result as for a single one since the destination STA is picked at random in every transmission.

4.3 Complex inter-WLAN interactions

In order to validate the behavior of Komondor in front of more complex inter-WLAN interactions, we now focus on the three-WLANs scenarios shown in

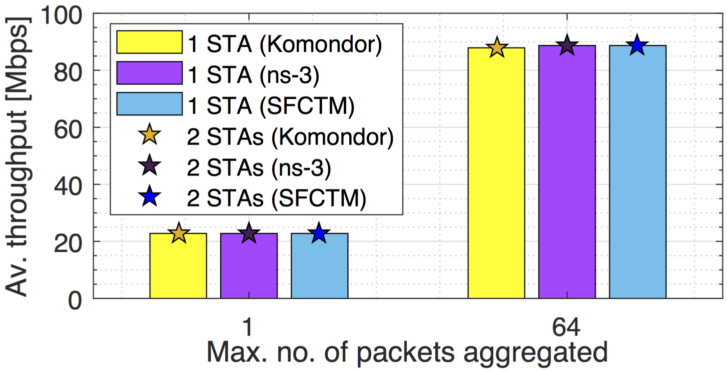
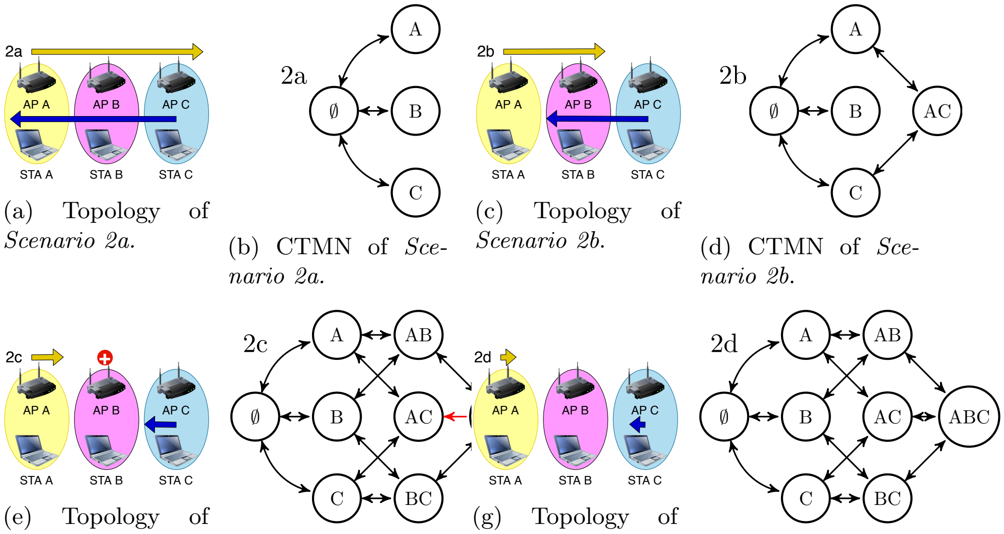


Figure 5: Average throughput experienced by the WLAN of *Scenario 1*, for $N_{\text{agg}} = 1$ and $N_{\text{agg}} = 64$. Results obtained from each simulation tool are shown.



(a) Topology of *Scenario 2a.*

(b) CTMN of *Scenario 2a.*

(c) Topology of *Scenario 2b.*

(d) CTMN of *Scenario 2b.*

(e) Topology of *Scenario 2c.*

(f) CTMN of *Scenario 2c.*

(g) Topology of *Scenario 2d.*

(h) CTMN of *Scenario 2d.*

Figure 6: Topologies and corresponding CTMNs of scenarios 2a-2d. The yellow and blue arrows represent the area of interference from transmitters in WLANs A and C, respectively, whereby medium contention is forced.

Fig. 6. We name them *Scenario 2a-2d*. The interactions occurring in such scenarios are illustrated through CTMNs, where states[6] represent the WLANs that are currently transmitting. Note that each of these scenarios reflects different situations that are of particular interest since they generalize different

[6] Note that CTMN states are not related by any means to Komondor states.

well-known phenomena in wireless networks:

- **Fully overlapping (Fig. 6a)**: all the nodes cause contention to all the others when transmitting. For that, the distance between consecutive APs and between AP and STA of the same WLAN is set to $d_{\text{AP,AP}} = d_{\text{AP,STA}} = 2$ m, respectively.
- **Flow starvation (Fig. 6c)**: contention is triggered in a pair-wise manner, so that WLAN_A and WLAN_C do not interfere each other. For that, the distance is set to $d_{\text{AP,AP}} = 4$ m and $d_{\text{AP,STA}} = 2$ m. Note that this case could be also extended to show a hidden node effect if AP_A or AP_C were intended to transmit to a STA located at the location of AP_B.
- **Potential overlap (Fig. 6e)**: contention only occurs at WLAN_B when both WLAN_A and WLAN_C transmit concurrently. Otherwise, the channel is sensed as free. Note that, in this case, packets are successfully transmitted in WLAN_B whenever it access the channel. The distances are $d_{\text{AP,AP}} = 5$ m and $d_{\text{AP,STA}} = 2$ m for WLAN_A and WLAN_C, and $d_{\text{AP,STA}} = 3$ m for WLAN_B.
- **No overlapping (Fig. 6g)**: none of the nodes causes contention to any other when transmitting. That is, every WLAN operates like in isolation. The distances in this case are $d_{\text{AP,AP}} = 10$ m and $d_{\text{AP,STA}} = 2$ m.

The average throughput experienced by each WLAN in each scenario is shown in Fig. 7. As previously done, we compare the performance of Komondor with ns-3 and SFCTMN. Note that results gathered by both Komondor and ns-3 are very similar in all the cases. Concerning the differences in the average throughput values estimated by both simulators and SFCTMN, we observe two phenomena with respect to backoff collisions in topologies of *Scenario 2a* and *2c*. First, in *2a*, the throughput is slightly higher when the capture effect condition is ensured. This is due to the fact that concurrent transmissions (or backoff collisions) are permitted and captured in the simulators. Second, the most notable difference is given in *2c*, which is caused by the assumption of continuous time backoffs in the CTMN. These are clear examples of the limitations of the analytical tool.

4.4 High-density and simulator performance

Finally, we assess the performance of Komondor when dealing with high-density scenarios. Notice that being able to simulate scenarios with a large number of nodes is a key feature due to the ever-increasing trend towards short-range and dense deployments. In this situation, we show the results of different fully-overlapping scenarios, ranging from 1 to 50 WLANs, each consisting in of one AP and one STA. The validation is performed against the Bianchi's analytical model and ns-3. The MCS for all the WLANs is set to 256-QAM. Fig. 8 shows the results in terms of throughput (average and aggregate) and collision probability obtained for fully overlapping networks of different sizes. For comparison

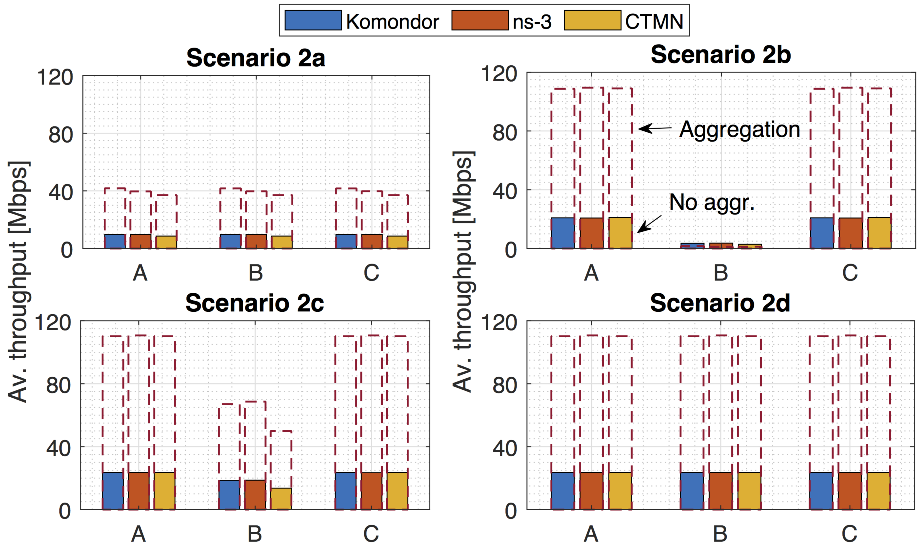


Figure 7: Average throughput experienced by each WLAN in scenarios 2a-2d. $N_{\text{agg}} = 1$ and $N_{\text{agg}} = 64$ are represented through solid bars and dashed lines, respectively.

purposes, the simulation time used in each scenario has been set to 100 seconds, for both Komondor and ns-3. Notice that such a fully overlapping setting frames a worst-case situation regarding packet collisions. This impacts on the number of events and the simulation time as the network density increases. Nevertheless, much more positive results are expected to be achieved in more realistic non-fully overlapping dense scenarios.

As shown, Komondor maintains its accuracy with respect to Bianchi's model, even when dealing with a lot of nodes. Regarding ns-3, slight differences are noticed in the collisions probability due to the error rate model, where collisions are based on the dropped RTS frames and the use of the Extended Interframe Space (EIFS). Moreover, differences in the throughput increase with the number of nodes, as previously addressed in [26].

To conclude this section, we provide insights into the execution complexity of Komondor. Fig. 9 shows the execution time and the number of generated events in Komondor and ns-3 for each number of WLANs.[7] As shown, the execution complexity of ns-3 is significantly higher than in Komondor. We identify the cause of this difference to be the complex PHY implementation in ns-3, which leads to a larger number of generated events.

[7] Note that the execution time is strongly dependent of the computer used and its status at the moment of performing the simulation. In our case, we used an Intel Core i5-4300U CPU @ 1.9 GHz x 4 and 7.7 GiB memory.

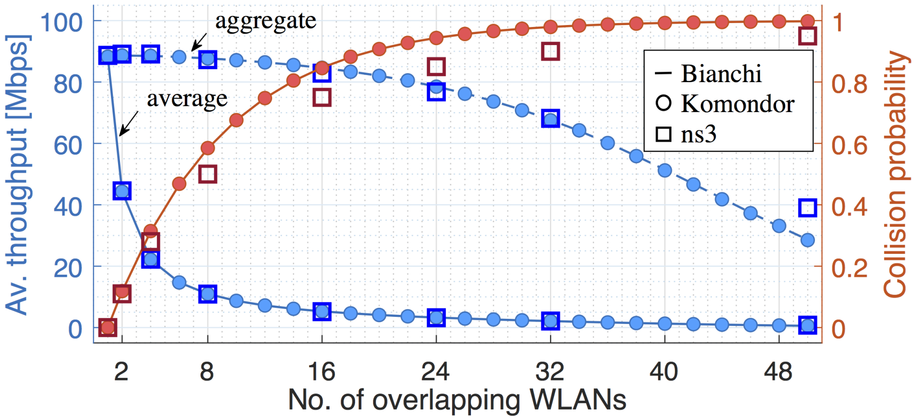


Figure 8: Throughput (average and aggregate) and collision probability vs. number of overlapping WLANs. Only some ns-3 points are plotted for the sake of visualization.

5 Komondor and potential use cases

Apart from small deployments consisting of few WLANs under single-channel operation [27], more complex scenarios capturing DCB or high-density scenarios have been already validated and analyzed by using Komondor. In this section, we briefly discuss further potential uses such as the implementation of next-generation WLAN techniques or the inclusion of learning agents to perform efficient spectrum access and spatial reuse.

5.1 Potential usage

Complex wireless environments can be already extensively simulated by Komondor as a result of its reduced computational complexity in comparison to other well-known simulators such as ns-3. A prominent example of a complex scenario mixing both high-density deployments and DCB is discussed in [21], where authors assessed the performance of different DCB policies versus node density (see Fig. 1). In [22], a similar deployment is analyzed while considering different traffic loads. A set of scenarios including DCB is shown in Fig. 10, which were validated in Komondor's validation report v0.1.[8] New features like spatial reuse, MIMO, beamforming and MU communications through OFDMA and/or MU-MIMO are currently under development.

[8] Komondor's validation report v0.1: https://github.com/wn-upf/Komondor/blob/master/Documentation/Other/validation_report_v01.pdf.

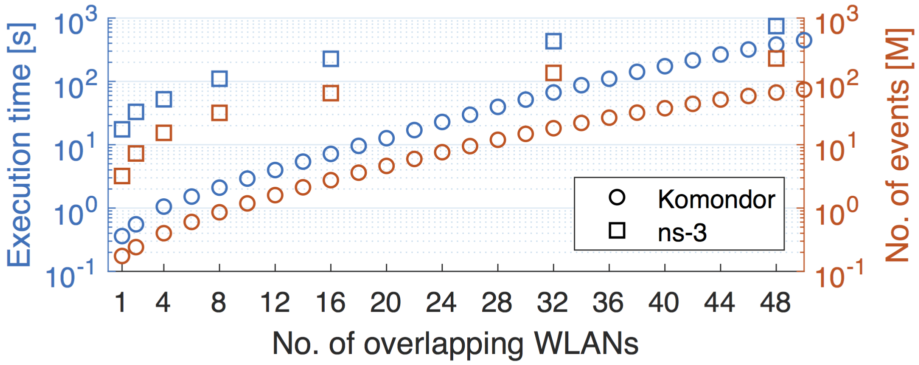


Figure 9: Execution time and number of generated events vs. number of overlapping WLANs.

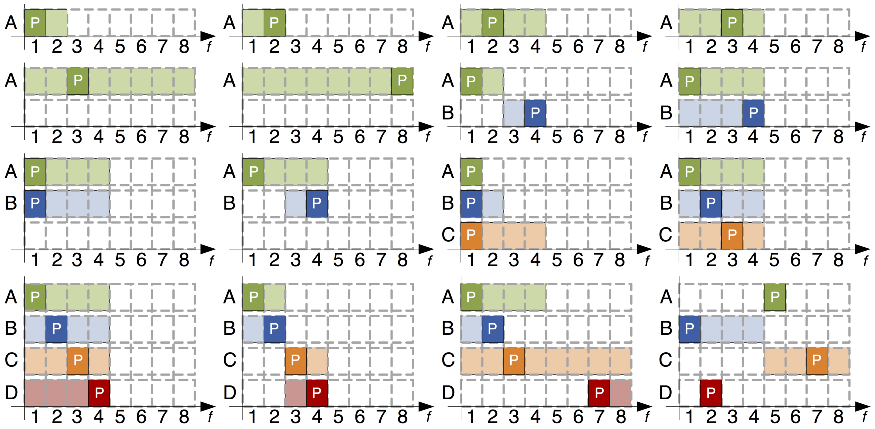
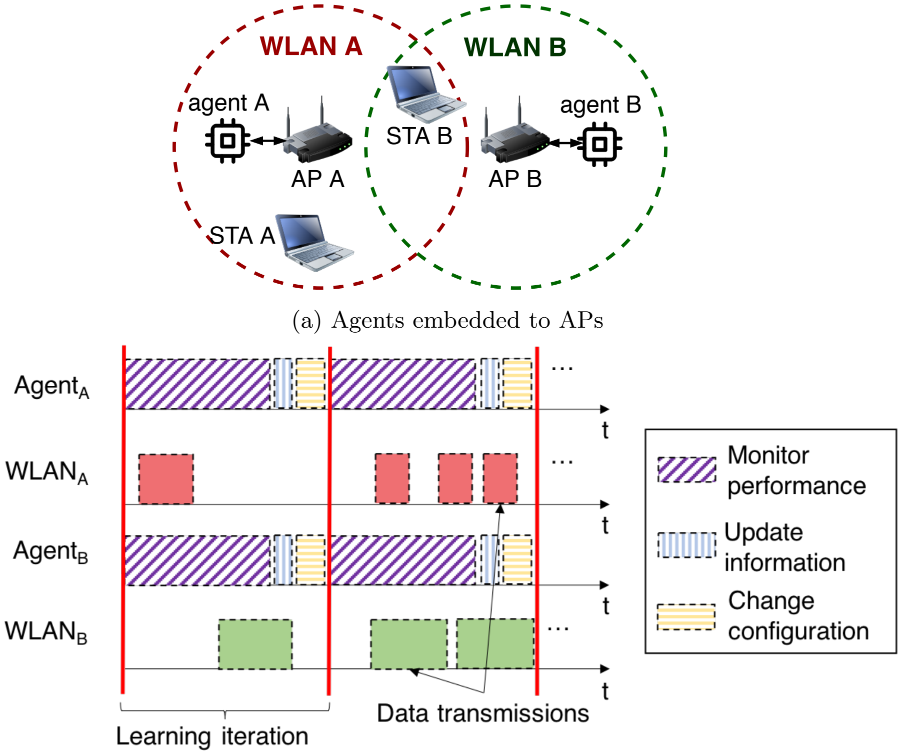


Figure 10: Scenarios with different DCB capabilities.

5.2 Machine learning agents

In addition to simulating advanced techniques proposed by the latest IEEE 802.11 amendments, Komondor permits including intelligent agents. In particular, agents are embedded to APs (see Fig. 11a) to perform the following operations (see Fig. 11b): *i)* monitor WLAN's performance, *ii)* run an implemented learning method, and *iii)* suggest new configurations to be applied by the WLAN, according to generated knowledge.

The application of intelligent agents has been previously studied in [27, 28], where decentralized learning is employed to both Transmit Power Control (TPC) and Carrier Sense Threshold (CST) adjustment.



(a) Agents embedded to APs

(b) Learning operation followed by agents

Figure 11: ML-based operation implemented in Komondor.

6 Conclusions

In this work, we presented Komondor, a wireless network simulator that stems from the need of providing a reliable and low-complexity simulation tool able to capture the operation of novel WLAN mechanisms like DCB or spatial reuse. The operation of Komondor has been validated against the ns-3 simulator and analytical tools such as CTMNs and Bianchi's DCF model. In this regard, we have shown its effectiveness when dealing with high-density scenarios, thereby outperforming ns-3 with respect to the simulation time. The provided validation is fundamental for the next development stages, which contemplate the inclusion of novel techniques in WLANs that have not been fully implemented in other well-known simulators. Some future implementations contemplate OFDMA, MU-MIMO, and the spatial reuse operation, naming a few among others. Finally, we have discussed the potential of Komondor regarding complex scenarios and ML integration. In particular, a preliminary ML-based architecture is already implemented, so that intelligent agents can rule self-configuring operations at different communication levels.

Acknowledgments

This work has been partially supported by a Gift from CISCO University Research Program (CG#890107) & Silicon Valley Community Foundation, by the Spanish Ministry of Economy and Competitiveness under the Maria de Maeztu Units of Excellence Programme (MDM-2015-0502), and by the Catalan Government under grant SGR-2017-1188. The work by S. Barrachina-Muñoz is supported by an FI grant from the Generalitat de Catalunya.

References

[1] I. Selinis, K. Katsaros, M. Allayioti, S. Vahid, and R. Tafazolli, "The Race to 5G Era; LTE and Wi-Fi," *IEEE Access*, vol. 6, no. 1, pp. 56 598–56 636, December 2018.

[2] "IEEE p802.11ax/d2.0," pp. 1–596, November 2017.

[3] B. Bellalta, "IEEE 802.11ax: High-efficiency WLANs," *IEEE Wireless Communications*, vol. 23, no. 1, pp. 38–46, February 2016.

[4] Y. Ghasempour, C. R. da Silva, C. Cordeiro, and E. W. Knightly, "IEEE 802.11 ay: Next-generation 60 GHz communication for 100 Gb/s Wi-Fi," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 186–192, 2017.

[5] L. C. et al., "EXtreme Throughput (XT) 802.11," 2018.

[6] G. Chen and B. K. Szymanski, "Reusing simulation components: cost: a component-oriented discrete event simulator," in *Proceedings of the 34th conference on Winter simulation (WSC): exploring new frontiers.* Winter Simulation Conference, 2002, pp. 776–782.

[7] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops.* ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, p. 60.

[8] A. K. Rathi and A. J. Santiago, "The new NETSIM simulation program," *Traffic engineering & control*, vol. 31, no. 5, 1990.

[9] J. Milos, L. Polak, M. Slanina, and T. Kratochvil, "Link-level simulator for WLAN networks," in *Link-and System Level Simulations (IWSLS), International Workshop on.* IEEE, 2016, pp. 1–4.

[10] T. Issariyakul and E. Hossain, "Introduction to Network Simulator 2 (NS2)," in *Introduction to Network Simulator NS2.* Springer, 2012, pp. 21–40.

[11] G. F. Riley and T. R. Henderson, "The ns-3 network simulator," in *Modeling and tools for network simulation.* Springer, 2010, pp. 15–34.

[12] P. Gawłowicz and A. Zubow, "ns3-gym: Extending openai gym for networking research," *arXiv preprint arXiv:1810.03943*, 2018.

[13] R. Stacey *et al.*, "Proposed TGax draft specification," *IEEE 802.11-16/0024rl*, 2016.

[14] "Wi-Fi design," https://www.nsnam.org/docs/models/html/wifi-design.html, 2018.

[15] R. Stacey, "Specification framework for tgax," https://mentor.ieee.org/802.11/dcn/15/11-15-0132-17-00ax-spec-framework.docx, 2016.

[16] Cisco, "NS3 Simulator of 802.11ax (ns3-802.11ax-simulator)," {https://github.com/cisco/ns3-802.11ax-simulator}, year=2017.

[17] G. Pei and T. R. Henderson, "Validation of OFDM error rate model in ns-3," *Boeing Research Technology*, pp. 1–15, 2010.

[18] I. Selinis, M. Filo, S. Vahid, J. Rodriguez, and R. Tafazolli, "Evaluation of the DSC Algorithm and the BSS Color Scheme in Dense Cellular-like IEEE 802.11ax Deployments," in *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Sept 2016, pp. 1–7.

[19] I. Selinis, K. Katsaros, S. Vahid, and R. Tafazolli, "Exploiting the Capture Effect on DSC and BSS Color in Dense IEEE 802.11 ax Deployments," in *Proceedings of the Workshop on ns-3.* ACM, 2017, pp. 47–54.

[20] E. Khorov, A. Kureev, I. Levitsky, and A. Lyakhov, "Testbed to Study the Capture Effect: Can We Rely on this Effect in Modern Wi-Fi Networks," in *2018 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, June 2018, pp. 1–5.

[21] S. Barrachina-Munoz, F. Wilhelmi, and B. Bellalta, "Dynamic Channel Bonding in Spatially Distributed High-Density WLANs," *(forthcoming) IEEE Transactions on Mobile Computing*, 2019.

[22] S. Barrachina-Muñoz, F. Wilhelmi, and B. Bellalta, "To Overlap or not to Overlap: Enabling Channel Bonding in High-Density WLANs," *Computer Networks*, vol. 152, pp. 40 – 53, 2019.

[23] S. Merlin, et al., "TGax Simulation Scenarios," doc. IEEE 802.11-14/0980r16, 2016.

[24] B. Bellalta, A. Zocca, C. Cano, A. Checco, J. Barcelo, and A. Vinel, "Throughput analysis in csma/ca networks using continuous time markov networks: a tutorial," in *Wireless Networking for Moving Objects.* Springer, 2014, pp. 115–133.

[25] 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, 2000.

[26] R. Patidar, S. Roy, T. R. Henderson, and M. Mehrnoush, “Validation of wi-fi network simulation on ns-3,” 2017.

[27] F. Wilhelmi, S. Barrachina-Muñoz, B. Bellalta, C. Cano, A. Jonsson, and G. Neu, “Potential and pitfalls of multi-armed bandits for decentralized spatial reuse in wlans,” *Journal of Network and Computer Applications*, vol. 127, pp. 26–42, 2019.

[28] F. Wilhelmi, C. Cano, G. Neu, B. Bellalta, A. Jonsson, and S. Barrachina-Muñoz, “Collaborative spatial reuse in wireless networks via selfish multi-armed bandits,” *Ad Hoc Networks*, 2019.

Table 2: Parameters considered in the presented scenarios.

Parameter	**Description**	**Value**
f_{c}	Central frequency	5 GHz
$\lvert c \rvert$	Basic channel bandwidth	20 MHz
MCS	11ax MCS index	0-11
G_{tx}	Transmitting gain	0 dB
G_{rx}	Reception gain	0 dB
$\mathrm{PL}(d)$	Path loss (Residential scenario)	see [23]
N	Background noise level	-95 dBm
σ_{leg}	Legacy OFDM symbol duration	4 µs
σ	OFDM symbol duration (GI-32)	16 µs
N_{sc}	Number of subcarriers (20 MHz)	234
N_{ss}	Number of spatial streams	1
T_{e}	Empty slot duration	9 µs
T_{SIFS}	SIFS duration	16 µs
T_{DIFS}	DIFS duration	34 µs
T_{PIFS}	PIFS duration	25 µs
$T_{\mathrm{PHY\text{-}leg}}$	Legacy preamble duration	20 µs
$T_{\mathrm{HE\text{-}SU}}$	HE single-user field duration	100 µs
T_{ACK}	ACK duration	28 µs
T_{BACK}	Block ACK duration	32 µs
$T_{\mathrm{PPDU}}^{\mathrm{max}}$	Max. PPDU duration	5484 µs
$L_{s,l}$	Size OFDM symbol (legacy)	24 bits
L_{D}	Data packet size	11728 bits
N_{agg}	No. of frames in an A-MPDU	1, 64
L_{RTS}	Length of an RTS packet	160 bits
L_{CTS}	Length of a CTS packet	112 bits
L_{SF}	Length of service field	16 bits
L_{MH}	Length of MAC header	320 bits
CW	Contention window (fixed)	15