Spectrum-Sensing-Based WiFi Performance Emulator for Experimental Evaluation of Cognitive Solutions

David Plets[1], Mostafa Pakparvar[1], Kris Vanhecke[1], Wout Joseph[1], *Senior Member, IEEE*, Luc Martens[1], *Member, IEEE*

(email:david.plets@intec.UGent.be phone:+32 9 33 14918)

[1]Department of Information Technology, Ghent University/iMinds, Ghent, Belgium

***Index Terms*— networking and QoS, traffic and performance monitoring, living lab, QoS, WiFi, cognitive, radio environment map, emulator, spectrum, sensing**

I. Context and Objectives

As the wireless medium is becoming more and more congested, a better use of the scarce frequency spectrum is needed to deliver the required Quality of Service (QoS) to clients. Especially in unlicensed bands, wireless systems suffer from intra-network interference [1], e.g., in professional wireless conferencing environments [2]. Therefore, cognitive solutions are being developed, both on radio and networking level [3, 4]. A way of approaching cognitive radio is by relying on spectrum sensing within the considered environment and building a real-time Radio Environment Map (REM), which visualizes Received Signal Strength Indicator (RSSI) values, power values, or even Mean Opinion Scores (MOS) scores. If nodes in the environment could be configured themselves, the REM and its corresponding output data can be used to test cognitive decision engine (CDE) interference scenarios. This paper will describe a system that allows experimentally emulating WiFi traffic in an actual (pseudo-shielded) environment and visualizing the related QoS and QoE (Quality of Experience) parameters. The emulator allows testing cognitive corrective actions of a CDE.

The outline of the paper is as follows. In Section II, the architecture of the system and its different components are discussed. Section III describes the three emulation scenarios that are executed. Section IV discusses the emulation results and finally, Section V summarizes the conclusions of this paper.

This work was supported by the iMinds-ICON QoCON project, co-funded by iMinds, a research institute founded by the Flemish Government in 2004, and the involved companies and institutions.

II. System Architecture

Fig. 1 shows the architecture of the emulator. It uses a network controller to generate the desired traffic in the considered network. The network controller commands are either manually generated by the user (through the user interface of the system), or either automatically, by a CDE. All network information (throughput, measured signal power, delay, jitter,…) is logged in a database which serves as a base to build the REM. The REM visualizes the network state to the user through the user interface and leads to new decisions (either by the user or by the CDE). When the network controller is configured by the CDE, the system can be considered as a cognitive networking system. When it is configured by the user, the system is in ‘emulation mode’. In this paper, the use of the emulator will be illustrated based on the execution of different scenarios. We will now discuss the different components of the system architecture.

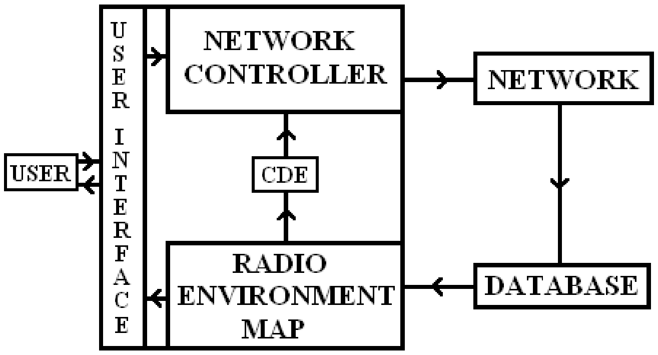


Fig. 1: System architecture.

A. *Network*

The considered **network** (see Fig. 2) is a pseudo-shielded testbed environment (w-iLab.t) in Ghent, Belgium. It consists of 60 nodes, mounted in an open room (66 m x 20.5 m) in a grid configuration with an x-separation of 6 m and a y-separation of 3.6 m. Fig. 2 shows the ground plan of the living lab with an indication of the location of the nodes (blue).

Each node has two WiFi interfaces (Sparklan WPEA-110N/E/11n mini PCIe 2T2R chipset: AR9280) and to each WiFi card, two antennas are connected (2x2 MIMO is supported). Furthermore, an RM090 sensor node and a USB 2.0 Bluetooth interface (Micro CI2 - v3.0 EDR) are incorporated into each node.

B. *Database*

As traffic is being generated, all network information (e.g., RSSI, delay, channel overlapping,…) is monitored and stored in a **database** (see Fig. 1). The database is a MySQL server running on the testbed server. Each second, new records are stored with their corresponding time stamp, technology, node ID, and frequency channel. The database also comprises a table with detailed information for all node IDs including physical location and available technologies (e.g. Wi-Fi, ZigBee, etc.) associated with the node.

C. *Network Controller and CDE*

The **network controller** (see Fig. 1) is a Java backend that passes the corresponding commands from the CDE or the user interface on to the target network elements (nodes) to steer their controllable parameters (transmission power, transmission rate, frequency channel, etc.) accordingly. The dissemination of the commands is facilitated by a custom hand shaking protocol between the nodes and the network controller. The network controller is steered by either a CDE (see Fig. 1) or either, as in this case, by the user via the user interface (see Fig. 1). Fig. 3 shows a screenshot of the user interface that connects to the network controller. The user interface allows configuring each node as sender node (indicated by outgoing arrow, e.g., node 22) to a certain receiver node (indicated by an incoming arrow, e.g., node 23), as a monitoring node (indicated by a black circle around the node, e.g., node 12) or as an inactive node (grey node, e.g., node 11) . For each sending node, the traffic parameters can be set: channel, protocol, data size or duration, packet size, data rate, and the number of parallel streams.

In future research, a **CDE** will be developed and implemented into the framework of Fig. 1. This will automate the cognitive corrective actions, based on the REM data.

D. *Radio Environment Map*

Based on the information in the database, a **radio environment map** (REM) (see Fig. 1) of the environment is created, visualizing Quality of Service (QoS) and Quality of Experience (QoE) parameters (e.g., throughput) or interference (e.g., channel occupancy) metrics. Since the number of sensing devices in the testbed is limited, the REM visualizer has to use interpolation techniques to estimate the values at other locations on the map. Various techniques could be used, but here Inverse Distance Weighting (IDW) or Shepard's algorithm is used to interpolate the data on the map. Fig. 4 shows an example of a REM, where the measured received power in the environment is visualized for a random transmit configuration .

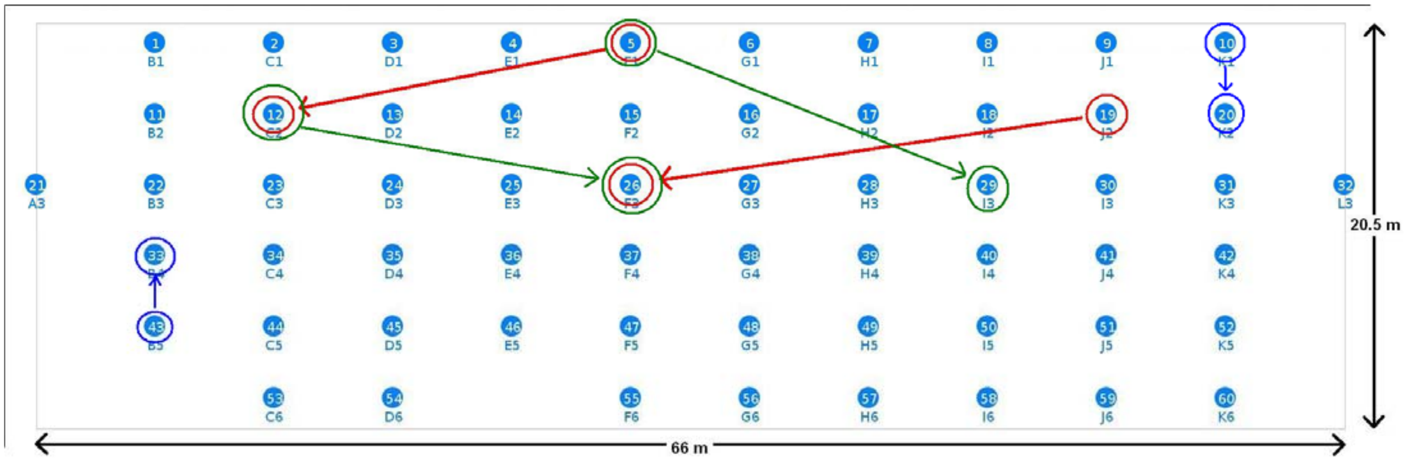
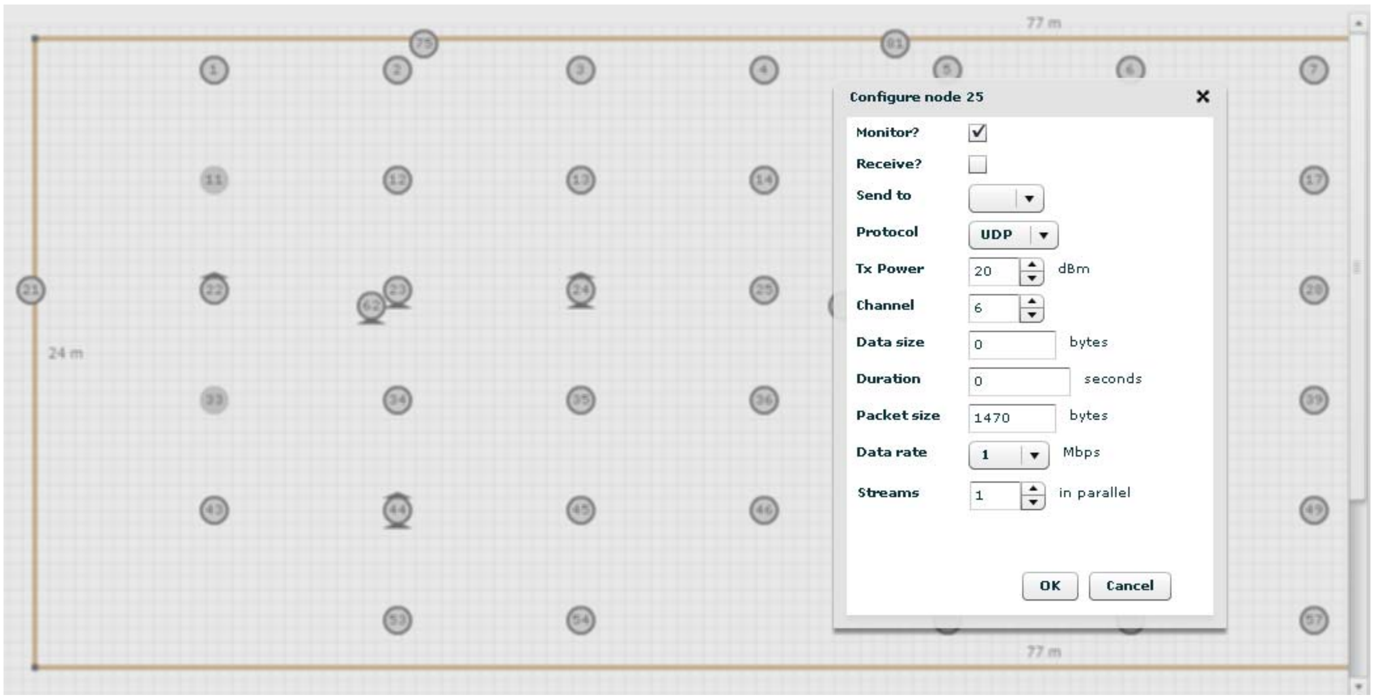


Fig. 2: W-iLab.t living lab test environment (66m x 20.5m) with indication of the node locations and the two configured links for scenario 1.



Configure node 25

Monitor?	✓
Receive?	
Send to	
Protocol	UDP
Tx Power	20 dBm
Channel	6
Data size	0 bytes
Duration	0 seconds
Packet size	1470 bytes
Data rate	1 Mbps
Streams	1 in parallel

Fig. 3: Network controller user interface for configuration of network nodes.

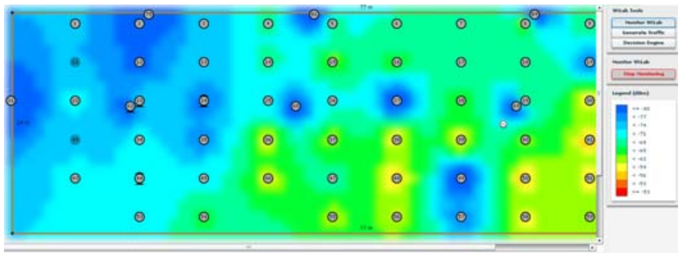


Fig. 4: A sample REM of the test-bed for the measured received power.

III. Emulation scenarios

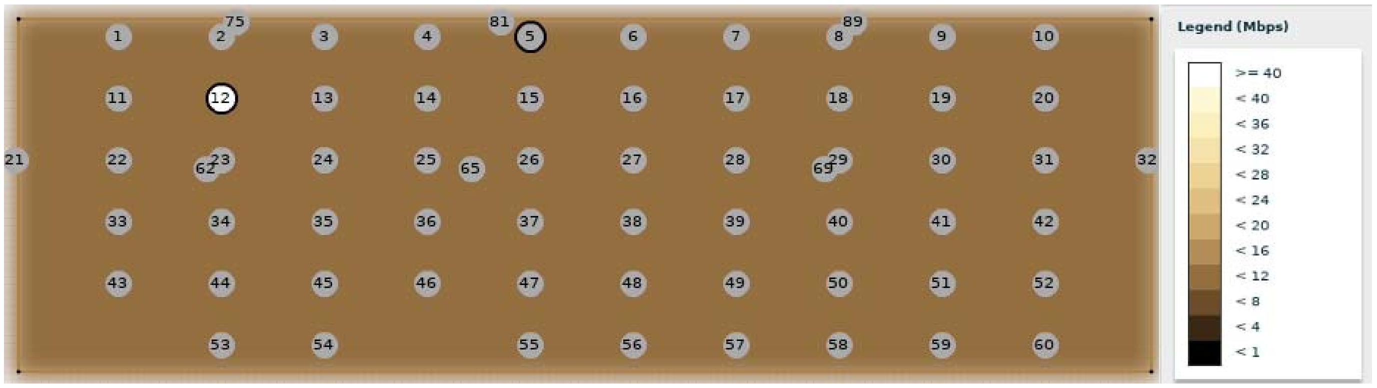
Three scenarios will be emulated and discussed. These scenarios all emulate possible corrective actions made by a CDE.

1. *Frequency adaptation:* Two links will be set up on the same channel and throughput values will be recorded. The user will then change the frequency of one link and assess the impact on the throughput of both links.
2. *Channel occupancy*: Two links will be set up, both with a low transmit rate of the radio interface (high channel occupancy). It will be assessed how both throughputs will be impacted when the interfaces' transmit rates are increased (lower channel occupancy). Channel occupancy COD is defined as the data generation rate DGR divided by the interface transmit rate TxRate.
3. *Transmit power*: Two short, distant links will be set up on the same channel where both transmitters have a high transmit power. It will be assessed how both throughputs are impacted when the transmit powers are lowered for interference reduction between the links.

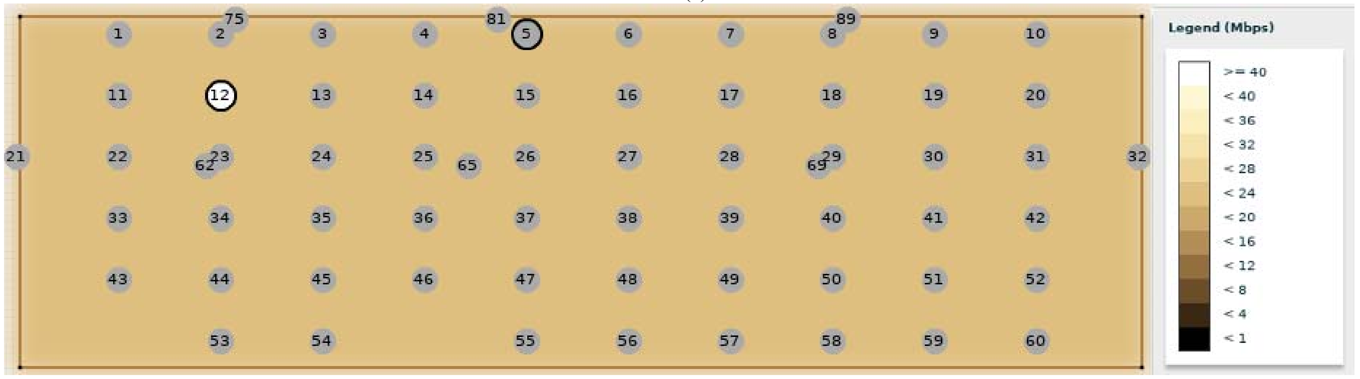
IV. Results

A. Frequency adaptation

In the first scenario, nodes 5 and 19 are configured to transmit packets to nodes 12 and 26 respectively, as indicated in Fig. 2 (red arrows). The links are both configured at channel 6, with a txRate (bit rate of radio interface [1]) of 54 Mbps and a channel occupancy degree (ratio of data generation rate and txRate [1]) of 100 %. Packets are transmitted during 60 s with a transmit power of 20 dBm. The resulting throughput on the links is only 11.5 Mbps on average, with a jitter of 3 ms. After emulating the cognitive corrective action, where link 'node 5 to node 12' switches its frequency to channel 11, the average **throughput** on both links **increases** to 21 Mbps **(+82.6%)** and the jitter reduces to 1.5 ms (-50%). Fig. 5 shows the throughput map due to node 5 (a) before and (b) after the channel switch. It clearly shows the QoS improvement due to the frequency adaptation. The figures also show that at each receiving location, the throughput is the same. This is due to open environment in the testbed. The throughput maps indicate the ease of assessing the impact of corrective actions on the QoS in the network.



(a)



(b)

Fig. 5: Throughput map due to node 5 before (a) and after (b) the channel switch.

B. *Channel occupancy*

In the second scenario, nodes 5 and 12 transmit packets to nodes 29 and 26 respectively, as indicated in Fig. 2 (green arrows). The transmit nodes operate at a txPower of 20 dBm. Initially, both links are configured to have a fixed COD of 100% (TxRate = DGR = 18 Mbps). The initial throughput on the link between 5 and 29 (link L1) equals 6.7 Mbps. A possible corrective action could be to lower the COD of L1 by increasing the TxRate to 54 Mbps. This corresponds to a COD of 33.33%. Execution of this corrective action causes a **throughput increase** to 9.0 Mbps (**+35.6%**), again showing the QoS improvement due to this corrective action. The emulator yields homogeneous throughput maps comparable to those in Fig. 5, but with different throughput values.

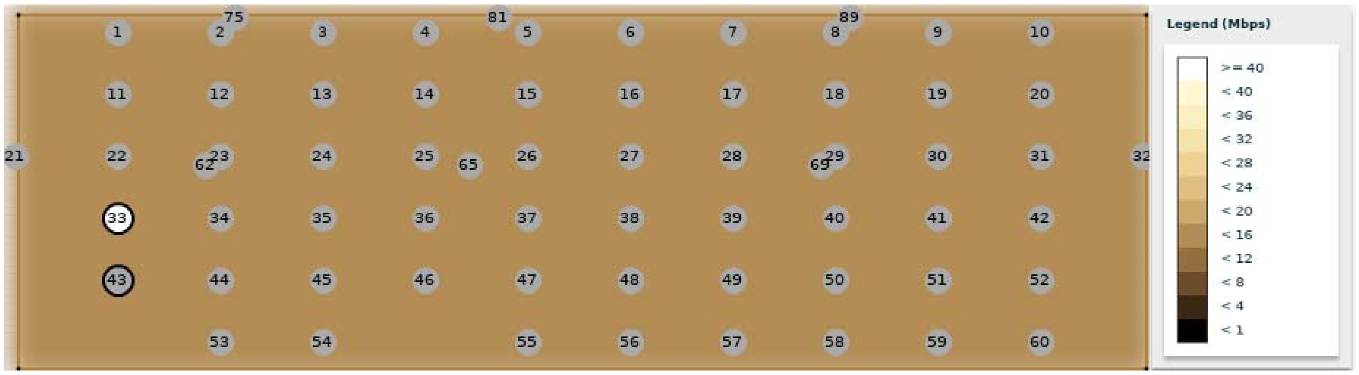
When both links are initialized with a TxRate and DGR of 11 Mbps, the throughput on L1 is 3.7 Mbps. Increasing the TxRate of L1 to 54 Mbps (COD decreases from 100% to 20.37%), makes the **throughput increase** to 5.7 Mbps (**+52.6%**).

C. *Transmit power*

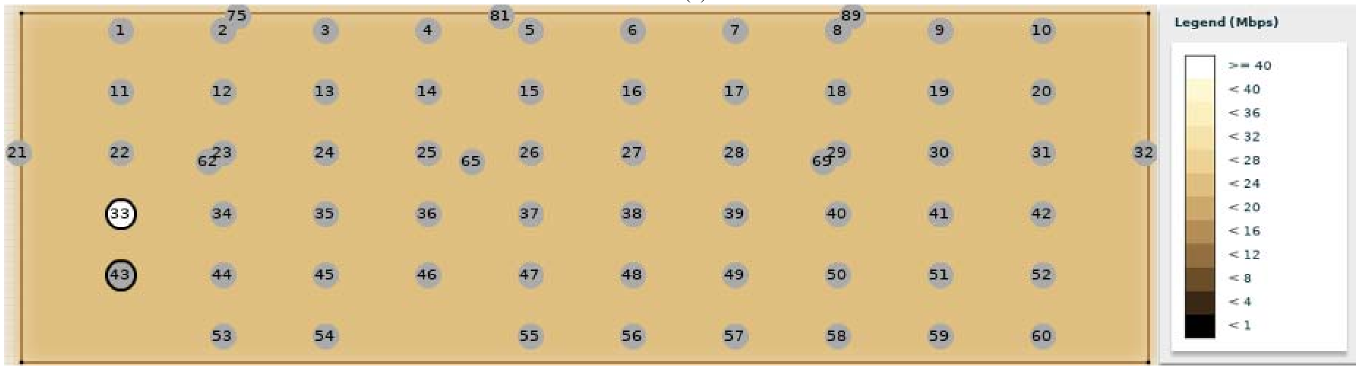
In the third scenario, two short and distant links are configured: node 43 transmits to node 33 (L1) and node 10 to node 20 (L2), as indicated in Fig. 2 (blue arrows). Initially, both links operate at full transmit power (20 dBm) and with a TxRate and DGR of 54 Mbps (COD = 100%). The throughput for both links then equals 14.5 Mbps. When the transmit power is lowered to 0 dBm for both links, the **throughput** on L1 **increases** to 22.6 Mbps (+**55.9%**), but **decreases** to 0.8 Mbps (**-94.9%**) on L2.

Fig. 6 (a) shows the throughput map due to node 43 before lowering the transmit power. The same figure is valid for the throughput map due to node 10 (14.5 Mbps). Figs. 6 (b) and 6 (c) show the throughput maps due to nodes 43 and 10 respectively, after lowering the transmit powers of these nodes from 20 to 0 dBm. The maps clearly show the improvement on L1 and the severely reduced throughput on L2.

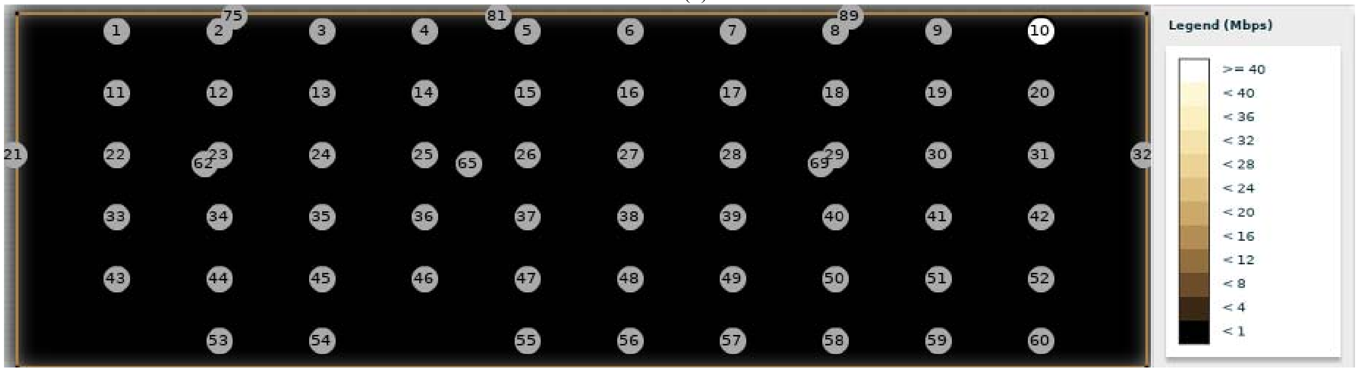
Further analysis with intermediate transmit powers delivers the results shown in Fig. 7. The figure shows that, as the power of the interference between the two links is reduced, the fairness declines and QoS on one of the links is indeed severely impaired. This is due to the Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) mechanism working worse in case of the lower transmit powers.



(a)



(b)



(c)

Fig. 6: Throughput map of (a) links L1 and L2 before lowering the transmit power, (b) L1 after lowering transmit power, and (c) L2 after lowering transmit power.

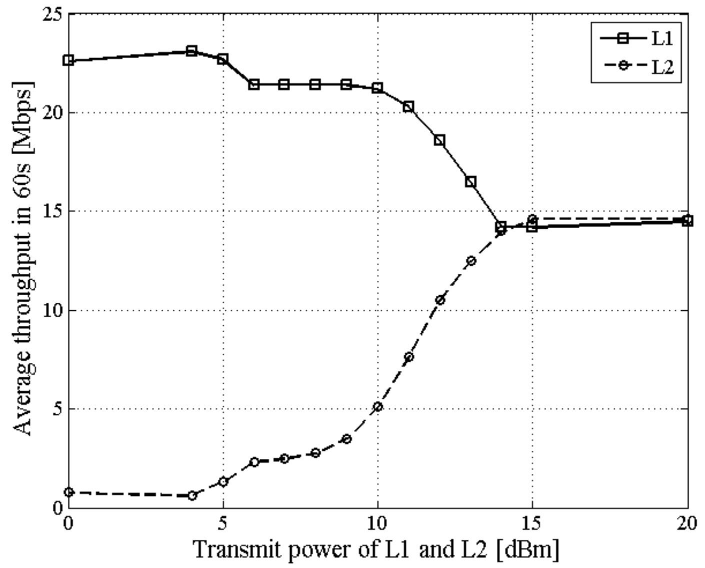


Fig. 7: Throughput over links L1 and L2 for varying transmit powers of the transmitting nodes.

V. Conclusions

In this paper, a spectrum-sensing-based WiFi performance emulator is presented. It allows assessing the impact of cognitive corrective actions with respect to QoS or QoE parameters (throughput, MOS score,…). The framework is designed to easily allow the implementation of various cognitive decision engine modules as well, leading to an automated cognitive networking system. Three emulation scenarios have been defined and the results are discussed, demonstrating the impact of the emulated cognitive corrective action. Adapting the frequency and decreasing the channel occupancy allows increasing throughput over the link-under-test, but lowering the transmit power on the other hand, can severely impact QoS. The proposed emulator allows testing different scenarios in a very user-friendly way. In the future, also other metrics will be visualized (latency, MOS score, …) and more advanced decision mechanisms for the execution of corrective actions will be tested.

VII. References

[1] D. Plets, M. Pakparvar, W. Joseph, L. Martens, “Influence of Intra-Network Interference on Quality of Service in Wireless LANs”, 2013 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), London, UK, 5-7 June, 2013, pp. 1-5.

[2] F. Heereman, W. Joseph, E. Tanghe, D. Plets, and Luc Martens, “Path Loss Model and Prediction of Range, Power and Throughput for 802.11n in Large Conference Rooms”, International Journal of Electronics and Communications, Int. J. Electron. Commun. (AEÜ), vol. 66, no. 7, pp. 561-568, 2012.

[3] Mitola, J., III; Maguire, G.Q., Jr.; , "Cognitive radio: making software radios more personal," Personal Communications, IEEE , vol.6, no.4, pp.13-18, Aug 1999

[4] Quer, G.; Meenakshisundaram, H.; Tamma, B.; Manoj, B.S.; Rao, R.; Zorzi, M.; , "Cognitive Network Inference through Bayesian Network Analysis," Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE , vol., no., pp.1-6, 6-10 Dec. 2010