A Low-Cost Large-Scale Framework for Cognitive Radio Routing Protocols Testing

Ahmed Saeed and Mohamed Ibrahim
Wireless Research Center
Egypt-Japan University for Sci. and Tech.
Alexandria, Egypt
Email: {ahmed.saeed,mibrahim}@ejust.edu.eg

Khaled A. Harras
Computer Science Department
School of Computer Science
Carnegie Mellon University Qatar
Email: kharras@cs.cmu.edu

Moustafa Youssef
Wireless Research Center
Egypt-Japan Univ. of Sc. and Tech. and
Alexandria University, Alexandria, Egypt
Email: {moustafa.youssef}@ejust.edu.eg

***Abstract*—Cognitive radio networks (CRNs) provide a solution to increase the utilization of the scarce radio frequency spectrum. Building testbeds for CRNs is one of the main challenges that can affect the wide deployability of such networks. In this paper, we present the design, implementation, and evaluation of *CogFrame*: a framework that facilitates the development of cost-efficient large-scale CRNs routing protocols testbeds. The framework allows the designers to focus on the CRNs routing protocols by abstracting the PHY and MAC layers while providing the necessary cross layer functionalities. *CogFrame* works with standard computers and WiFi cards to reduce the cost while allowing integration with other special hardware for more flexibility. In addition, *CogFrame* provides different modules for implementing and emulating complex scenarios such as regulatory authority policies, mobility management, and topology management.**

We benchmark the performance of *CogFrame* and compare it to standard ns-2 simulations and USRP2 implementations. In addition, we case study a location-aided routing protocol for CRNs using both *CogFrame* and ns-2 simulations. Our results highlight the ease of implementation, low-cost, and realistic replication of the CRN environment, showing the promise of *CogFrame* as a testbed for future CRNs implementations.

I. Introduction

The exponential demand for portable and mobile devices increased the demand for high bandwidth wireless communications. Given that the spectrum is a limited natural resource, traditional static spectrum allocation mechanisms lead to the waste of this valuable resource. *Cognitive radios* emerged as a solution for enhancing spectrum utilization by allowing unlicensed users to opportunistically utilize unused portions of the spectrum. Despite the vast progress made in advancing the different research directions in cognitive radio networks (CRNs), (e.g. spectrum sensing, spectrum management, and routing [1], [2]), most of the evaluations made for that work are either through simulation or on small-scale testbeds in controlled environments [3], [4]. Simulations provide a tool for testing large-scale CRNs. However, it is usually difficult for them to capture realistic environment conditions, especially given the dynamic nature of CRNs [5], [6]. This observation is even more important in the case of CRNs, where convincing the licensed spectrum holders to share it with the secondary users (SUs) requires real-life demonstrations, rather than simulations.

On the other hand, real testbeds [4] have focused on the flexibility of designing and evaluating new PHY and MAC protocols/solutions for which a small testbed requiring special hardware, such as USRP and WARP boards, is sufficient. Examples of large-scale open access testbeds [7]–[9] offer both PHY and MAC layers prototyping by using GNU Radio over USRPs connected to general purpose hosts. The authors in [10] proposed using WARPs connected to cloud services. This was demonstrated by connecting the WARPs to powerful PCs using MATLAB to easily configure the WARPs. Another approach is the development of emulation testbeds [11]; The BEE2 FPGA is used as an emulation board to connect RF-frontends representing primary users and secondary users. The FPGA helps emulate the different possible scenarios for communication. Similarly, the SORA architecture [12] presents a new software and hardware stack that addresses increasing the processing power dedicated to PHY and MAC operations. Although hardware-based approaches have advantages for prototyping PHY and MAC layer protocols, they share a number of drawbacks: (1) the prohibitive monetary cost of a large-scale testbed, (2) shared testbeds do not scale to a large number of users and do not allow private deployment, and (3) there is a lot of complexity added to the development and deployment of routing protocols due to the requirement of implementing the whole protocol stack and handling PHY and MAC issues.

From a different perspective, software architectures have been proposed to ease the development of MAC and PHY protocols for CRNs. Hydra [13] is a flexible wireless network testbed that eases the development of MAC and PHY protocols based on combining the flexibility of Click Router for **MAC** and GNU Radio for PHY using USRPs as RF-frontend. Iris [14] main goal is runtime reconfiguration, enabling seamlessly changing of protocol modules based on observations in the traffic. Other work provides a library of reusable code and an easy way to write customized code aiming at easing rapid development of a CRN stack [15]. While these approaches provide good frameworks for developing a CRN stack, they still require building the **whole stack and configuring it**. In addition, all of them require special RF-frontends to support software defined radio libraries.

In this paper, we present *CogFrame*, a framework that enables low-cost large-scale testing of cognitive radio routing protocols. *CogFrame* is designed to work on standard computers with off-the-shelf WiFi cards. It builds on the Click modular router [16] and extends it to provide an abstraction layer that supports the development of new routing protocol for CRNs, including cross-layer functionalities such as spectrum sensing and spectrum management. *CogFrame* can also interface with special hardware (such as USRP boards), in a transparent way to the developer, if needed (e.g. to integrate with other PHY and MAC protocols). Moreover, *CogFrame* provides emulation modules for modeling complex scenarios such as primary user (PU) behaviors, PU and SU mobility patterns, and topology management. In addition, an integral part of *CogFrame* is a Policy Manager module that allows capturing different radio policies enforced by regulatory authorities or restricting the operation channels. The rest of this paper is organized as follows: we present our design goals and the details of the *CogFrame* framework in Section II. Section III benchmarks *CogFrame* and compares it to USRPs and ns-2 simulations. We present a case study of using *CogFrame* to implement a recent location-aided routing protocol for CRNs (LAUNCH [17]) in Section IV. Finally, Section V concludes the paper and gives directions for future work.

II. The CogFrame Routing Framework

In this section, we introduce the design and implementation of *CogFrame*. We start with our design goals followed by the architecture description.

A. *CogFrame Design Goals*

CogFrame is designed with the following goals in mind:

1) Flexible Cross-Layer Interactions: A CR node needs to be able to sense and acquire information on the state of the spectrum. Also, it needs to be able to specify its transmission parameters based on this sensed state. While not all transmission parameters need to be handled by the routing protocol, some of these parameters (e.g. channel used and transmission power) might need to be controlled. This requires cross-layer interactions. However, it should not require the development and maintenance of the whole CR stack by the routing protocol designer. The testbed architecture should support this interaction putting a minimal burden on the routing protocol designer in terms of PHY and MAC layer handling.

2) Complex Scenario Implementation and Emulation: Testing a routing protocol within a realistic environment with factors including realistic channel conditions, topology, mobility of both PUS and SUs, and complex PU behavior is one of the main goals of developing a testbed for CRNs. These conditions are hard to achieve on conventional hardware-based testbeds (e.g. USRPs). Methods that facilitate implementing, synthesizing, and emulating these conditions are necessary to provide realistic results for the developed testbed.

3) Low Code Development Overhead: Another cost that should be taken into consideration is that of code development for building the testbed. A modular testbed with reusable reconfigurable components that could be ported, easily-tested,

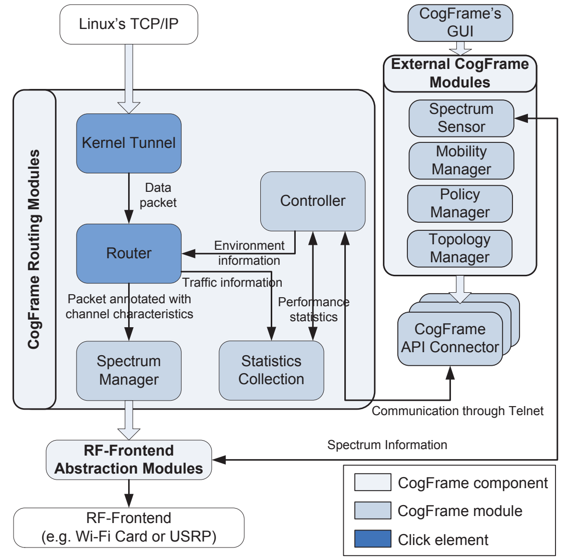


Fig. 1. *CogFrame* Architecture.

and reconfigured would reduce the cost of the testbed development and enrich the framework's community by providing module libraries. The developer of the routing protocol should not be involved in coding the MAC and PHY layer.

4) Low Cost Large Realistic Experiments: The final challenge is providing a low monetary cost testbed. Conventional testbeds rely on using special purpose custom-developed hardware. The cost of building a large scale testbed with this special purpose hardware could be prohibitive. Although there are available open access large scale testbeds (e.g. ORBIT [8], Emulab [9] and VT-CORNET [7]), the problem remains with the complexity of deployment and control of these testbeds, scaling to a large number of users, privacy, and the large cost of replicating the same experiment locally.

B. *CogFrame Architecture*

To support the above goals, we design *CogFrame* to work over cheap commodity hardware, leveraging PHY and MAC protocols of conventional WiFi cards. *CogFrame* builds on top of the Click modular router, inheriting its modularity, and provides APIs and libraries for common CRNs functionalities such as channel switching and spectrum sensing as well as complex scenario implementation and emulation. In addition, it provides a flexible modular structure that allows the integration of special purpose hardware (e.g. USRP and WARP boards) to improve the flexibility of the PHY and MAC layers if needed.

Figure 1 shows *CogFrame* architecture. It is composed of three main components: 1) Routing modules, 2) External modules and 3) RF-frontend abstraction modules. We now provide the details of each of the components

1) Routing Modules: CogFrame routing modules are built on top of the Click modular router [16]. Click provides simple modules with limited functionalities called elements. Elements are connected together to form a directed graph configuration that represents the path of the packet through the router. The Click architecture is extensible, enabling protocol designers to build their own elements and configure graphs representing the different functionalities of a routing protocol.

Routing protocol designers using *CogFrame* will still write Click elements for their protocol. However, instead of dealing with the details of MAC and PHY layers, *CogFrame* provides the following Click elements that abstract the interaction between the routing protocol and these layers and abstracts an easy-to-use cross-layer APIs:

• *Spectrum Manager*: this element expects a packet annotated by the channel it is supposed to be sent on and the transmit power, then configures the Wi-Fi interface to work on the specified channel and the specified power. This is done by interacting with the RF-frontend Abstraction module (Section II-B3).

• *Controller*: this element uses the element handlers feature of the Click router to enable other programs interface with the router to either query its state or modify its parameters. The *Controller* provides handlers to allow other modules (Section II-B2) that are responsible for spectrum sensing, mobility tracking, policy change detection, etc to communicate with the router; separating these functionalities from the routing implementation. The information obtained through the handlers is then transferred to the routing elements to make routing and other channel selection decisions.

• *Statistics Collector*: this element is used by the protocol designer to collect information on spectrum utilization and traffic patterns. This module helps provide insight on the performance of the implemented protocol in terms of total throughput and spectrum utilization.

2) External Modules: These modules are not part of the Click router but are responsible for providing functionalities that are required by a CRN testbed. These include spectrum sensing (including PU detection), mobility management, policy management, and topology management. This information can either be obtained from the physical hardware on the device (such as GPS or WiFi cards) or emulated to support complex scenarios. The modules are:

• *Spectrum Sensor*: provides information about the state of the spectrum sensed by the Wi-Fi card including available channels, channel quality, etc. This can be performed by active or passive scanning [18]. In addition, this module can also provide different PU emulation scenarios.

• *Mobility Manager*: which is responsible for informing the router about the current node position. This is useful in scenarios like location-aided routing protocols as well as helping other modules, such as the topology and policy managers.

• *Policy Manager*: provides the router with the operation constraints to ensure compliance to the regularity rules. These constraints include, e.g., the TV white space available channels in a certain location obtained by contacting the geo-location database [19]. In addition, it can be used to limit the available channels to avoid interference with nearby APs.

• *Topology Manager*: is responsible for enforcing certain network topologies on the participating nodes by emulating different channel qualities on individual links to neighboring nodes. This is particularly useful for emulating topologies that span large geographic areas and/or poor channel conditions.

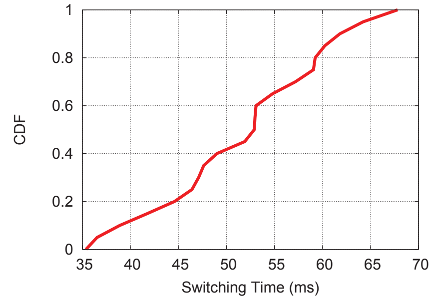
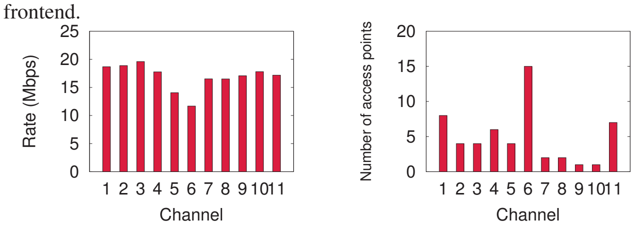


Fig. 2. CDF of switching time for *CogFrame* when using a Wi-Fi RF-frontend.



(a) Maximum throughput (b) Num. of APs

Fig. 3. Maximum throughput analysis achieved by *CogFrame* in an 802.11g settings in a university dorms.

3) RF-frontend Abstraction Modules: This module is responsible for abstracting the functionality of the PHY and MAC layers by providing an API to handle the spectrum management and spectrum sensing functionalities that the routing protocol requires while hiding the exact implementation of these functions for a certain RF-frontend. *CogFrame* natively supports WiFi cards by using `ioctl` commands to control the WiFi card. It also supports power control and scanning operations to sense the presence of other SUs or PUs. This module also allows a modular extension to *CogFrame* to support other hardware, such as USRP and WARP boards.

4) Other Modules: *CogFrame* comes with a GUI that eases the configuration of different external components as well as monitoring the status of the framework. The *CogFrame* API connector allows the external modules to communicate with the router through a control socket using the telnet protocol, as specified by the Click router. This separates the design and the implementation of the router from the sensory modules.

III. CogFrame Benchmark

In this section, we benchmark the performance of packet transmission on *CogFrame*, in terms of channel switching time and throughput, while using conventional Wi-Fi cards as an RF-frontend, and compare it to both USRPs and ns-2 simulations. We study the performance of a typical protocol in the next section.

To obtain these results, we send packets using *CogFrame* implemented on two Lenovo G570 laptops with Atheros AR9285 802.11abgn wireless LAN cards configured to work with 802.11g[1]. Similar performance was observed for other bands (i.e. 802.11abn). This experiment was conducted in the university dorms.

[1]This leads to only three non-overlapping channels: 1, 6, and 11.

Framework	Max. Throughput (Mbps)	Switch. Time (ms)	Cost ($)	Development overhead
USRP N200 [20]	56	5	Machines cost + 1500 **per node**	High
ns-2 [21]	User Defined	User Defined	One machine	Low
CogFrame	56	52.9	Machines cost	Low

TABLE I
COMPARISON BETWEEN *CogFrame*, USRP2 AND NS-2 SIMULATIONS ON WI-FI.

A. Switching Time

For switching time measurement, the benchmark makes the two nodes agree on channel 11 then switch to either channel 6 or channel 1. Figure 2 shows the CDF of *CogFrame* switching time. The figure shows that the median switching time is $52.9ms$ which conforms to typical WiFi channel switching times. This can be further enhanced to around $5ms$ if needed by engineering the switching process as in [22].

B. Throughput

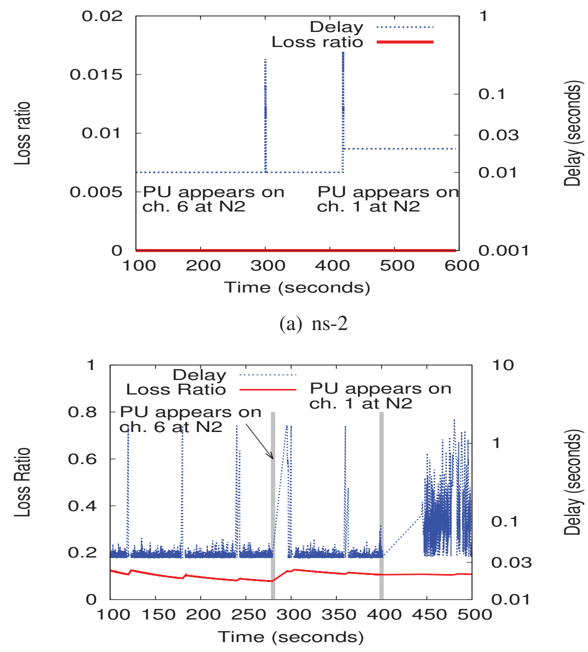
For maximum throughput, the benchmark router sends packets with the MTU with a rate of $56Mbps$. Figure 3(a) shows the maximum achievable throughput on each channel. The loss in data rate can be due to the collision with existing APs on the same channels. Figure 3(b) shows the number of APs seen on each channel. The figures show that there is a correlation between the throughput on each channel and the number of access points. There are exception though, e.g. Channel 1, as the APs working on this channel are far away from the testbed and hence have a lower effect on throughput. This is compared to Channel 6, which had the strongest AP. This correlation has been confirmed by studying the traffic distribution of each AP. Note also that, due to the overlapping between different channels in WiFi, APs on certain channels affect the traffic on nearby channels, as in the case of Channel 6.

C. Comparison with Other Evaluation Methods

Table I compares the different operational parameters of *CogFrame*, *USRP N200* [20], and ns-2 simulations [21]. Although *USRP N200* supports large bandwidth and high transmission power, this requires the development of special MAC and PHY layers, which is prohibitive for a routing protocol designer. Therefore, we believe that using the available 802.11 building blocks for USRP will usually be the easier choice, making the available bit rate for Wi-Fi RF-frontends and the USRPs similar. Still, in the USRP case, the routing protocol designer will have to incur the disadvantages of USRPs cost, development, and deployment overheads. As for ns-2 simulations, although a high bit rate is supported, the bit-level simulation nature of ns-2 makes this computationally prohibitive. Therefore, users usually use ns-2 with low data rates of 2 Mbps. In addition, parameters used in the simulations are usually user defined and a lot of relaxing assumptions regarding transmission parameters and node processing capabilities are made making the results deviate from reality [5], [6].

IV. CASE STUDY

In this section, we compare the performance of a recently proposed location-aided routing protocol (LAUNCH [17]) for CRNs using both ns-2 simulations and *CogFrame*. Our goal is to highlight the differences between the two evaluation methods and argue for the practicality of *CogFrame*.



(a) ns-2

(b) *CogFrame*

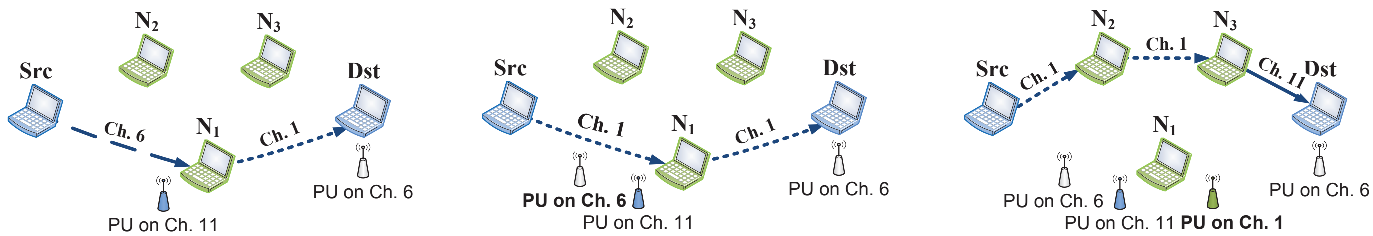
Fig. 4. The performance of LAUNCH on ns-2 and *CogFrame* in terms of delay (log-scale) and loss ratio.

A. LAUNCH in CogFrame

We choose LAUNCH as it requires three RF-frontends demonstrating *CogFrame*'s cost efficiency and scalability. In addition, it uses a number of *CogFrame* components, e.g. the mobility manger (for location information) and spectrum manager (for PU emulation). We implemented LAUNCH's architecture over *CogFrame*, on a testbed of five Lenovo G570 laptops equipped with built-in Atheros AR9285 802.11abgn cards for the sending RF-frontend, and TP-Link TL-WN723N USB Wi-Fi cards as the receiving RF-frontend (implementing the locking mechanism of LAUNCH), and an ethernet card as the common control channel interface. Note that the developer needs **only to implement two modules** to realize LAUNCH as all the remaining modules belong to either *CogFrame* or Click.

B. Performance Comparison

Figure 5 shows a scenario that we used to test LAUNCH's performance in terms of delay and loss ratio using both ns-2 simulations and the developed *CogFrame* testbed. The scenario includes changing behavior of different PUs leading to changing the channel at time 280s and the entire path at time



(a) The source selects the node closest to the destination (N_1) as a next hop on Channel 6. N_1 in turn selects Channel 1 to communicate with the destination.

(b) A PU appears on Channel 6 at N_1 at time 280s. Channel between source and N_1 is changed to Channel 1.

(c) Another PU appears on Channel 1 at N_1 at time 400s. The source is forced to choose a new route to the destination through N_2 then N_3.

Fig. 5. *LAUNCH* [17] case study.

420s. Both experiments were made using the same parameters using 802.11g as the MAC layer and 300 Kbps as the data rate (to match [17] simulation parameters). Figures 4(a) and 4(b) show the results. The figure shows that the simulations have almost no loss ratio and minimal delay. On the other hand, *CogFrame* presents a more realistic outcome, where real channel conditions and dynamics lead to frequent delays and losses. Also, the effect of channel switching and path switching (due to PU activity) is evident on both loss ratio and delay. These results highlight that simulation can be far from reality and that *CogFrame* can be used to efficiency and quickly implement CRNs routing protocols.

V. Conclusion

We presented *CogFrame* as a new configurable, cost efficient, flexible framework for the rapid development of CRNs routing protocols. The framework allows the protocol designer to focus on the design issues for the routing protocol by abstracting the MAC and PHY layers. It leverages the functionalities of standard computers and Wi-Fi cards, saving the cost of special purpose RF-frontends while giving the flexibility for supporting other RF-frontends. In addition, it provides novel functionalities for supporting CRNs such as regulatory policy management, PU emulation, as well as traditional wireless network features such as mobility and topology management. We benchmarked the performance of *CogFrame* and showed it advantages over traditional ns-2 simulations and USRP2 boards. We also showed a case study of the ease of implementing a recent location-aided routing protocol on *CogFrame* and how simulations cannot capture the real-life problems.

Our future plans for *CogFrame* include enhancing the GUI to provide easier tools for developing the protocol with a drag-and-drop user interface, providing an interface to allow the interaction between ns-2 simulations and *CogFrame* testbeds for more realistic simulations, and extending the framework to support transport layer protocols for CRNs.

VI. Acknowledgment

This work is supported in part by a grant from the Egyptian NTRA. The authors would like to thank Fadel Digham for his feedback on the architecture.

References

[1] I. Akyildiz, W.-Y. Lee, M. Vuran, and S. Mohanty, “A survey on spectrum management in cognitive radio networks,” *Communications Magazine, IEEE*, vol. 46, no. 4, pp. 40 –48, april 2008.

[2] A. Abbagnale and F. Cuomo, “Gymkhana: a connectivity-based routing scheme for cognitive radio ad hoc networks,” in *INFOCOM*, 2010.

[3] P. Steenkiste, D. Sicker, G. Minden, and D. Raychaudhuri, “Future directions in cognitive radio network research,” *NSF Workshop Report*, 2009.

[4] K. Chowdhury and T. Melodia, “Platforms and testbeds for experimental evaluation of cognitive ad hoc networks,” *Communications Magazine, IEEE*, vol. 48, no. 9, pp. 96–104, 2010.

[5] D. Kotz, C. Newport, R. S. Gray, J. Liu, Y. Yuan, and C. Elliott, “Experimental evaluation of wireless simulation assumptions,” ser. MSWiM’04.

[6] S. Kurkowski, T. Camp, and M. Colagrosso, “MANET simulation studies: the incredibles,” *ACM MC2R*, vol. 9, no. 4, pp. 50–61, 2005.

[7] T. e. a. Newman, “Virginia tech cognitive radio network testbed and open source cognitive radio framework,” in *TridentCom’09*. IEEE, 2009, pp. 1–3.

[8] D. e. a. Raychaudhuri, “Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols,” in *IEEE WCNC’05*.

[9] E. Eide, L. Stoller, T. Stack, J. Freire, and J. Lepreau, “Integrated scientific workflow management for the emulab network testbed,” in *USENIX Annual Technical Conference*, 2006, pp. 363–368.

[10] Z. C. et al, “Towards a large-scale cognitive radio network: Testbed, intensive computing, frequency agility and security,” in *IEEE ICNC 2012*.

[11] S. e. a. Mishra, “A real time cognitive radio testbed for physical and link layer experiments,” in *IEEE DySPAN 2005*.

[12] K. e. a. Tan, “Sora: high performance software radio using general purpose multi-core processors,” ser. NSDI’09.

[13] K. Mandke, S. Choi, G. Kim, R. Grant, R. Daniels, W. Kim, R. Heath, and S. Nettles, “Early results on Hydra: A flexible MAC/PHY multihop testbed,” in *IEEE VTC2007-Spring*.

[14] P. Sutton, J. Lotze, H. Lahlou, S. Fahmy, K. Nolan, B. Ozgul, T. Rondeau, J. Noguera, and L. Doyle, “Iris: an architecture for cognitive radio networking testbeds,” *Communications Magazine, IEEE*, vol. 48, no. 9, pp. 114–122, 2010.

[15] C. e. a. González, “Open-source SCA-based core framework and rapid development tools enable software-defined radio education and research,” *Communications Magazine, IEEE*, vol. 47, no. 10, pp. 48–55, 2009.

[16] E. e. a. Kohler, “The click modular router,” *ACM Trans. Comput. Syst.*, vol. 18, no. 3, pp. 263–297, Aug. 2000.

[17] K. e. a. Habak, “A location aided routing protocol for cognitive radio networks,” in *IEEE ICNC 2013*.

[18] M. Youssef, L. Shahamat, M. Kleene, and A. Agrawala, “The ieee 802.11 active probing analysis and enhancements,” in *Wireless Networks, Communications and Mobile Computing, 2005 International Conference on*, vol. 2, june 2005, pp. 1539 – 1544 vol.2.

[19] Federal Communications Commission, “FCC Adopted Rules for Unlicensed Use of Television White Spaces,” Tech. Rep. ET Docket No. 04-186, Second Report and Order and Memorandum Opinion and Order, 2004.

[20] Ettus Research LLC. The USRP. [Online]. Available: http://www.ettus.com/

[21] K. F. S. McCanne, S. Floyd and K. Varadhan. The Network Simulator - ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns/

[22] S. Sengupta, K. Hong, R. Chandramouli, and K. Subbalakshmi, “Spiderradio: A cognitive radio network with commodity hardware and open source software,” *Communications Magazine, IEEE*, 2011.